

A multigrid perspective on the parallel full approximation scheme in space and time

Matthias Bolten¹, Dieter Moser^{2*}, Robert Speck²

¹*Department of Mathematics, Universität Kassel, Germany.*

²*Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Germany.*

SUMMARY

For the numerical solution of time-dependent partial differential equations, time-parallel methods have recently shown to provide a promising way to extend prevailing strong-scaling limits of numerical codes. One of the most complex methods in this field is the “Parallel Full Approximation Scheme in Space and Time” (PFASST). PFASST already shows promising results for many use cases and many more is work in progress. However, a solid and reliable mathematical foundation is still missing. We show that under certain assumptions the PFASST algorithm can be conveniently and rigorously described as a multigrid-in-time method. Following this equivalence, first steps towards a comprehensive analysis of PFASST using block-wise local Fourier analysis are taken. The theoretical results are applied to examples of diffusive and advective type. Copyright © 2016 John Wiley & Sons, Ltd.

Received ...

KEY WORDS: parallel-in-time; PFASST; multigrid; local Fourier analysis; high-performance computing

1. INTRODUCTION

Due to the rapid increase of the number of cores of todays and future HPC systems the demand for new parallelization strategies has grown rapidly in the last decades. When the speedup of a parallelization of the spatial dimensions is saturated, one general idea is to utilize parallelization of the temporal dimension. In [1] we find a classification of such methods, divided into parallelization *across the step*, *across the method* or *across the problem*.

Direct time-parallel methods mostly belong to the class of parallelization across the method, examples are certain parallel Runge-Kutta methods [2, 3]. Only modest parallel speedup is expected for these methods, because the number of processing units used for the parallelization are bound by e.g. the number of Runge-Kutta stage values. Other direct methods for parallel-in-time integration include RIDC [4], ParaExp [5], tensor-product space-time solvers [6] or methods using Laplace transformation [7].

If a method decomposes the problem into subproblems which are solvable in a parallel manner and couples these subproblems using an iterative method, it typically belongs to the class of parallelizations across the problem. The most prominent example are waveform relaxation methods [8, 9], which are part of the broad area of domain decomposition methods.

First ideas of parallel-in-time integration date back to Nievergelt in 1964 [10], which belongs to the class of multiple shooting methods and hence to the class of parallelizations across the step. More parallel-in-time integration methods were found in the area of multiple-shooting

*Correspondence to: E-mail: d.moser@fz-juelich.de

methods [11, 12]. Among them, in 2001 by Lions et al., *Parareal* [13] renewed the interest in parallel-in-time methods and sparked many new papers in its field. The success of Parareal is accounted to its simplicity and applicability: Only a fine but expensive and a coarse but cheap propagator in time have to be provided by the user. Then, parallelization across the temporal dimension can be achieved in an iterative prediction-correction manner. In principle, the number of processing units is not bounded, but depends on the actual decomposition of the time domain.

Parareal influenced other methods (see [14]) or even inspired the design of new methods. In [15], the Parareal approach is coupled to iterative solvers of a collocation problem, the so called spectral deferred correction (SDC) methods. This approach is extended to the “parallel full approximation scheme in space and time” (PFASST) in [16]. PFASST adopts and evolves the characteristics of Parareal by interweaving its iterations with those of the local SDC scheme. In addition, PFASST uses ideas from the theory of nonlinear FAS multigrid methods.

Multigrid methods in general have a long-standing successful history and a solid mathematical basis, see e.g. [17]. Regarding parallel-in-time integration, the first attempt using multigrid ideas dates back to Hackbusch in 1984 [18]. Since then, multigrid methods were further developed and resulted, e.g., in the multigrid waveform relaxation [19, 20], in multigrid reduction-in-time [21], or in classical space-time-multigrid [22, 23]. All these classes are not strictly separated from each other. Oftentimes methods may be reformulated to fit into a new class. A prominent example is Parareal itself: it was reformulated as a multiple shooting method as well as a multigrid method in [24], which in turn paved the way for a comprehensive analysis of Parareal.

This already shows the growing number and diversity of parallel-in-time methods. A classification of PFASST into the diversity of methods contributes to the understanding of PFASST by opening up the opportunity to use different mathematical tools from different fields. In particular, multigrid theory offers a variety of tools such as local Fourier analysis to estimate the convergence and to obtain a priori error bounds. A mathematical analysis becomes more and more important for the comparison of these algorithms and the design of algorithms for different applications.

The goal of this work is to associate PFASST with multigrid methods and apply the tools we find in multigrid theory along the lines of two standard problems, namely diffusion and advection. This sheds light on a general strategy how to estimate the convergence rate of PFASST and hereby the number of iterations needed to achieve a certain precision.

To achieve this goal, we proceed as follows: In Section 2.1, we introduce the notation and preliminaries necessary to state a matrix formulation of PFASST and its constituents. In particular, we introduce the collocation problem and the notation to deal with the nested multilevel structure of our setting. On this basis, we introduce the spectral deferred correction and its multi-level enhancement in matrix form in Section 2.2 and 2.3. Then, we introduce PFASST in algorithmic form in Section 2.4 which is then converted into matrix form in 3.1 to 3.3. This matrix form facilitates the use of ideas from multigrid analysis in Section 4 to 4.2, which leads to a block-decomposition of the iteration matrix of PFASST. In Section 5, we introduce four strategies to estimate the convergence rate of PFASST. The work is closed with an outlook and a conclusion in Section 6.

2. THE PARALLEL FULL APPROXIMATION SCHEME IN SPACE AND TIME

We start with a brief introduction of the building blocks of PFASST from the perspective of linear iterative solvers. To this end, we restrict ourselves to linear autonomous ordinary differential equations and—for the multi-level parts—to two levels only. We will comment on these restrictions in Section 6.

2.1. Preliminaries and Notation

The starting point is the linear autonomous ordinary differential equation in the Picard formulation

$$U(t) = U_0 + \int_{t_0}^t \mathbf{A} U(\tau) d\tau, \quad t \in [t_0, T], \quad (1)$$

where \mathbf{A} is a discretized spatial operator, e.g., stemming from a method of line discretization of a partial differential equation. For the discretization in the temporal dimension the time domain $[t_0, T]$ is divided into L subintervals. Each subinterval $[t_{l-1}, t_l]$, with $l \in \{1, \dots, L\}$, contains a set of M nodes $\{\tau_1, \dots, \tau_M\}$. We choose

$$0 = t_0 < \dots < t_L = T, \quad t_l < \tau_1 < \dots < \tau_M = t_{l+1}, \quad (2)$$

$$\Delta t = t_{l+1} - t_l, \quad \Delta \tau_m = \tau_{m+1} - \tau_m.$$

Each set of nodes $\{\tau_1, \dots, \tau_M\}$ are used as quadrature nodes for the numerical integration with rules like, e.g., Gauß-Radau or Gauß-Lobatto. Note that the last quadrature node coincides with the right border of the particular subinterval, which simplifies the formal notation of the algorithm. The results translate to other quadrature rules with minor modifications, though. Furthermore, if a mathematical entity like a set of numerical values or a certain matrix belongs to a subinterval $[t_l, t_{l+1}]$ we denote it e.g. by $\mathbf{U}_{[t_l, t_{l+1}]}$ (if it is not clear from the context).

Due to the nested structure and the distinct treatment of spatial and temporal dimensions, an appropriate notation is needed. Continuous functions are always represented by lower case letters, discretized and semi-discretized functions are the upper case version. Let $u(t, x)$ be a function in space and time, defined on the domain $[t_0, T] \times \mathbb{R}$, with $T \in \mathbb{R}_+$. For N degrees-of-freedom in space x_1, \dots, x_N we use the notation

$$\mathbf{U}(t) = (u(t, x_1), u(t, x_2), \dots, u(t, x_N))^T \in \mathbb{R}^N, \quad t \in [t_0, T]$$

for semi-discretization in space. A full space-time discretization is denoted as

$$\mathbf{U}_{[t_{l-1}, t_l]} = (U(\tau_1), U(\tau_2), \dots, U(\tau_M))^T \in \mathbb{R}^{M \cdot N}, \quad \tau_i \in [t_{l-1}, t_l], \quad l \in \{1, \dots, L\},$$

$$\mathbf{U} = (\mathbf{U}_{[t_0, t_1]}, \dots, \mathbf{U}_{[t_{L-1}, T]})^T \in \mathbb{R}^{M \cdot N \cdot L}.$$

On each subinterval a *collocation problem* is posed. It arises, when quadrature is used as a numerical counterpart to the integration in (1). The basis for most quadrature formulations is the interpolation, easily expressed using the Lagrange polynomial basis $\{\ell_i\}_{i=1}^M$, with

$$\ell_i(s) := \prod_{k=1, k \neq i}^M \frac{s - \tau_k}{\tau_i - \tau_k}. \quad (3)$$

If we weight each Lagrange polynomial with the evaluation of the function $f(t)$ at the point τ_i and sum them up, we get the interpolation polynomial of the function $f(t)$, which is exact on the nodes $\{\tau_1, \dots, \tau_M\}$. Now, quadrature is nothing more than using the exact integration values of the interpolation polynomial as approximations for the integration of $f(t)$. The following definition employs this strategy.

Definition 1. Let $a < \tau_1 < \tau_2 < \dots < \tau_M = b$ be the set of quadrature nodes and \mathbf{Q} the quadrature matrix with entries

$$q_{i,j} = \int_a^{\tau_j} \ell_i(\tau) d\tau, \quad i, j = 1, \dots, M.$$

We discretize (1) at the quadrature nodes, using the matrix \mathbf{Q} as approximation of the integral and obtain this set of linear equations:

$$U(\tau_i) = U(t_0) + \sum_{j=1}^M q_{i,j} \mathbf{A} U(\tau_j), \quad i = 1, \dots, M.$$

Using the Kronecker product and the vector of ones $\mathbf{1}_M \in \mathbb{R}^M$ we write this system of linear equations as

$$\mathbf{U} = \mathbf{U}_0 + \Delta t \mathbf{Q} \otimes \mathbf{A} \mathbf{U}, \quad \text{with } \mathbf{U}_0 = \mathbf{1}_M \otimes U(t_0),$$

or, more compactly,

$$\mathbf{M}\mathbf{U} = (\mathbf{I} - \Delta t \mathbf{Q} \otimes \mathbf{A}) \mathbf{U} = \mathbf{U}_0. \quad (4)$$

This problem is called **collocation problem** on $[a, b]$.

The set of quadrature nodes determines the kind of quadrature. Well-known quadrature rules are Chebyshev, Gauß-Legendre, Gauß-Radau, and Gauß-Lobatto. These quadrature rules have a spectral order, which is reflected in the high order of the numerical solution of the collocation problem. Gauß-Radau and Gauß-Lobatto quadrature rules use quadrature nodes which are in accordance with (2). Due of the higher order we focus on the Gauß-Radau quadrature rule in this paper.

Finally, the PFASST algorithm is working on a hierarchy of discretizations. As mentioned before, we focus on the two-level version with spatial coarsening only, i.e. PFASST is solving on a coarse and a fine level in space. For both levels a separate set of operators and value vectors is needed. The coarse level versions are simply denoted with a tilde, e.g. $\tilde{\mathbf{A}}$ is the coarse level version of \mathbf{A} .

2.2. Spectral Deferred Corrections

Instead of directly solving the collocation problem on a subinterval, the spectral deferred corrections method (**SDC**) utilizes a low-order method to generate an iterative solution that converges to the collocation solution \mathbf{U} . SDC was first introduced by Dutt et al. [25] as improvement of deferred correction methods [26]. In the last decade, SDC was accelerated with GMRES or other Krylov subspace methods [27], enhanced to a high-order splitting method [28–30], and found its way into the domain of parallel respectively time-parallel computing [31, 32], in particular within PFASST [15, 16].

Regarding the setting of this paper, we cast SDC as a *preconditioned Richardson iteration method* for the collocation problem as defined in Definition 1. This was pointed out earlier by various authors. For example in the work of Weiser et al. [33] this interpretation was used to optimize the convergence speed of SDC.

A general preconditioned Richardson iteration, noted as

$$\mathbf{U}^{k+1} = \mathbf{U}^k + \mathbf{P}^{-1}(\mathbf{c} - \mathbf{M}\mathbf{U}^k), \quad (5)$$

is fully described by the preconditioner \mathbf{P} , the system matrix \mathbf{M} , and the right-hand side \mathbf{c} of the linear equation under consideration. \mathbf{P} has to be easy to invert, while being an accurate alternative for the system matrix \mathbf{M} . The SDC method follows this approach by replacing the dense quadrature matrix \mathbf{Q} by a lower triangular matrix \mathbf{Q}_Δ . One simple way to generate a lower triangular matrix is to use the rectangle rule for quadrature instead of the Gauß-Radau rule. In [33] an LU decomposition of \mathbf{Q} provides a \mathbf{Q}_Δ which results in better convergence properties than the use of the simple rectangle rule while requiring the same computational effort.

The particular choice

$$\mathbf{P}_{\text{SDC}} = \mathbf{I} - \Delta t \mathbf{Q}_\Delta \otimes \mathbf{A}, \quad \text{and} \quad \mathbf{c} = (U(t_0), U(t_0), \dots, U(t_0))^T \in \mathbb{R}^{NM}, \quad (6)$$

then allows us to write SDC as preconditioned Richardson iteration with system matrix \mathbf{M} as defined in Def. 1 and where the right-hand side is given by the initial values $U(t_0)$ of the ODE spread on each node. If SDC is used on another subinterval than the first, the right-hand side consists of a numerical approximation of $U(t_l)$ spread on each node. In order to start the iteration an initial iteration vector \mathbf{U}^0 is needed. For SDC, the right-hand side is an apparent choice for a initial iteration vector. With these choices, one Richardson iteration is equivalent to one SDC sweep [33, 34]. The iteration matrix of SDC is simply given by

$$\begin{aligned} \mathbf{T}_{\text{SDC}} &= \mathbf{I} - \mathbf{P}_{\text{SDC}}^{-1} \mathbf{M} \\ &= \mathbf{I} - (\mathbf{I} - \Delta t \mathbf{Q}_\Delta \otimes \mathbf{A})^{-1} (\mathbf{I} - \Delta t \mathbf{Q} \otimes \mathbf{A}), \end{aligned} \quad (7)$$

Note that if we just use the lower triangular part of the \mathbf{Q} matrix as \mathbf{Q}_Δ , the preconditioned Richardson iteration is a Gauß-Seidel iteration. With \mathbf{Q}_Δ being a simpler integration rule or stemming from the LU decomposition of \mathbf{Q} instead of the lower triangular part of \mathbf{Q} , we characterize SDC as **approximative Gauß-Seidel iteration**.

2.3. Multi-level Spectral Deferred Corrections

The next step towards PFASST is the introduction of multiple levels in space (and time, which we will not consider here). This leads to a multi-level spectral deferred corrections method called (**MLSDC**), first introduced and studied in [35]. Here, SDC iterations (called “sweeps” in this context) are performed alternately on a fine and on a coarse level in order to shift work load to coarser, i.e. cheaper, levels. These cheaper levels are obtained, e.g., by reducing the degrees-of-freedom in space or the order of the quadrature rule in time. Therefore, MLSDC requires suitable interpolation and restriction operators \mathbf{T}_C^F and \mathbf{T}_F^C , and a coarse-grid correction in order to transfer information between the different levels. As a consequence, MLSDC can be written as a FAS-multigrid-like iteration. Like SDC it solves the collocation problem in an iterative manner, using the same initial iteration vector. For our purpose we derive a two-level version from [35] as:

1. Perform n_F fine SDC sweep using the values \mathbf{U}^k according to (5). This yields provisional updated values \mathbf{U}^* .
2. Sweep from fine to coarse:
 - (a) Restrict the fine values \mathbf{U}^* to the coarse values $\tilde{\mathbf{U}}^k$.
 - (b) Compute the FAS correction $\boldsymbol{\tau}^{k+1} = \tilde{\mathbf{M}}\tilde{\mathbf{U}}^k - \mathbf{T}_C^F \mathbf{M} \mathbf{U}^*$
 - (c) Perform n_C coarse SDC sweeps beginning with $\tilde{\mathbf{U}}^k$ and the FAS correction $\boldsymbol{\tau}^k$. This yields new values $\tilde{\mathbf{U}}^{k+1}$
3. Sweep from coarse to fine : Compute the interpolated coarse correction $\boldsymbol{\delta}^k$ and add it to \mathbf{U}^* to obtain \mathbf{U}^{k+1}

Note that we use the FAS correction strategy here to match the description of [35]. This is just a question of notation, because in the linear case using this correction strategy is equivalent to the standard coarse-grid correction [17]. Note further, that we will only perform one fine and one coarse SDC sweep in each MLSDC iteration, i.e. $n_F = n_C = 1$. The next lemma shows that we can cast this algorithm as a preconditioned Richardson iteration, too.

Lemma 1. Let $\mathbf{T}_C^F \in \mathbb{R}^{NM \times \tilde{N}\tilde{M}}$ and $\mathbf{T}_F^C \in \mathbb{R}^{\tilde{N}\tilde{M} \times NM}$ be the prolongation and restriction operators which transfer information between the coarse and fine level. We describe the same problem on a fine space-time grid with the system matrix \mathbf{M} and on a coarse space-time grid with $\tilde{\mathbf{M}}$. For both levels we use a preconditioned Richardson iteration method, which is characterized by \mathbf{P} and $\tilde{\mathbf{P}}$ to solve $\mathbf{M}\mathbf{U} = \mathbf{c}$ and $\tilde{\mathbf{M}}\tilde{\mathbf{U}} = \mathbf{T}_F^C \mathbf{c} = \tilde{\mathbf{c}}$, respectively. Then a combination of both methods using coarse-grid correction can be written as

$$\begin{aligned} \mathbf{U}^{k+\frac{1}{2}} &= \mathbf{U}^k + \mathbf{T}_C^F \tilde{\mathbf{P}}_{\text{SDC}}^{-1} \mathbf{T}_F^C (\mathbf{U}^0 - \mathbf{M}\mathbf{U}^k) \\ \mathbf{U}^{k+1} &= \mathbf{U}^{k+\frac{1}{2}} + \mathbf{P}_{\text{SDC}}^{-1} (\mathbf{U}^0 - \mathbf{M}\mathbf{U}^{k+\frac{1}{2}}) \end{aligned} \quad (8)$$

It is possible to write (8) in form of (5), using a new preconditioner $\mathbf{P}_{\text{MLSDC}}$, where

$$\mathbf{P}_{\text{MLSDC}}^{-1} = \mathbf{T}_C^F \tilde{\mathbf{P}}_{\text{SDC}}^{-1} \mathbf{T}_F^C + \mathbf{P}_{\text{SDC}}^{-1} - \mathbf{P}_{\text{SDC}}^{-1} \mathbf{M} \mathbf{T}_C^F \tilde{\mathbf{P}}_{\text{SDC}}^{-1} \mathbf{T}_F^C. \quad (9)$$

Following (7) and using \mathbf{P}_{SDC} and $\tilde{\mathbf{P}}_{\text{SDC}}$ yields the MLSDC iteration matrix

$$\mathbf{T}_{\text{SDC}} = \mathbf{I} - \left(\mathbf{T}_C^F \tilde{\mathbf{P}}_{\text{SDC}}^{-1} \mathbf{T}_F^C + \mathbf{P}_{\text{SDC}}^{-1} - \mathbf{P}_{\text{SDC}}^{-1} \mathbf{M} \mathbf{T}_C^F \tilde{\mathbf{P}}_{\text{SDC}}^{-1} \mathbf{T}_F^C \right) \mathbf{M}. \quad (10)$$

Proof

Let U^k be the result of the last iteration on the fine level. For the proof we start in the middle of the algorithm. First we compute the FAS correction

$$\tau^k = \tilde{M} T_F^C U^k - T_F^C M U^k \quad (11)$$

and use it to modify \tilde{c} for the next iteration on the coarse level. We start the iteration on the coarse level with

$$\tilde{U}^{k+1} = \tilde{U}^k + \tilde{P}_{\text{SDC}}^{-1} (\tilde{c} + \tau^k - \tilde{M} \tilde{U}^k) = T_F^C U^k + \tilde{P}_{\text{SDC}}^{-1} T_F^C (c - M U^k),$$

with the restricted value $\tilde{U}^k = T_F^C U^k$. Then, we compute the coarse correction

$$\delta^k = T_C^F (\tilde{U}^{k+1} - T_F^C U^k)$$

and obtain the half-step

$$U^{k+\frac{1}{2}} = U^k + \delta^k = U^k + T_C^F \tilde{P}^{-1} T_F^C (U^0 - M U^k)$$

after some algebraic manipulations. Using this half-step for the next iteration on the fine level gives (8). Simple algebraic manipulations, after inserting the half-step into the second step, yield the preconditioner (9), which immediately leads to the iteration matrix (10). \square

For the matrix formulation it is irrelevant whether the MLSDC step starts with the computation on the fine or the coarse level. To comply with the literature, we leave the algorithm of MLSDC in the original order, while changing the order for the matrix formulation.

As a part of PFASST, MLSDC corresponds to the computation performed on each subinterval. Adding a communication framework between the MLSDC iterations performed on each subinterval leads to PFASST. However, adding the communication framework yields a structure similar to the one we have seen in Lemma 1.

2.4. The PFASST algorithm

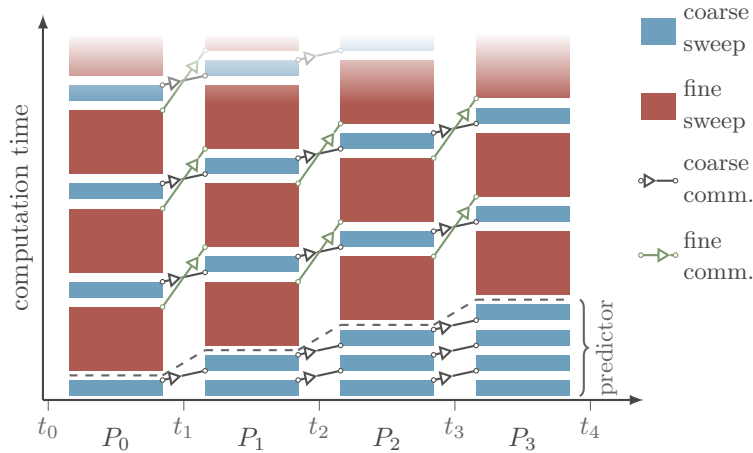


Figure 1. Schematic representation of the PFASST algorithm with two levels and four processes P_0, \dots, P_3 handling four parallel time steps. Created using `pfasst-tikz` [36].

The time-parallel PFASST algorithm in its final form was introduced in [16] as a combination of SDC methods [25] with Parareal [13] using an FAS correction strategy to allow for efficient spatial coarsening along the level hierarchy.

We explain PFASST on the basis of the schematic representation in Figure 1. First of all, we see the time domain, decomposed into subintervals, on the x-axis. On the y-axis we see the elapsed computational time. Each processor is assigned to a subinterval, where it performs MLSDC iterations and sends intermediate results on each level to the next processor. The blue and red blocks represent the SDC sweeps on the coarse and fine level. These blocks are connected through FAS corrections to the subjacent blocks (red to blue). The arrows represent the communication between the processors. Before starting with the actual PFASST iterations, a prediction phase, represented by the first blue blocks near the x-axis, computes suitable initial values for the iterations to come.

Based on the schematic representation and the full algorithm description in [16], we state a two-level version without the prediction phase. Let $U_{[t_{l-1}, t_l], m}^k$ be the value on the l -th subinterval at the k -th iteration and the m -th node. We have

$$\mathbf{F}_{[t_{l-1}, t_l]}^k = [\mathbf{A} U_{[t_{l-1}, t_l], 1}^k, \dots, \mathbf{A} U_{[t_{l-1}, t_l], M_l}^k] \text{ and } \mathbf{U}_{[t_{l-1}, t_l]}^k = [U_{[t_{l-1}, t_l], 1}^k, \dots, U_{[t_{l-1}, t_l], M_l}^k],$$

where M_l is the number of nodes on the l -th interval. An upper bar, e.g. \bar{U}_{l-1}^{k+1} , indicates that this value was sent by the preceding processor. These values are used as a new right-hand side to the collocation problem on the following subinterval. Denote the initial values for each subinterval as $U_{[t_{l-1}, t_l], m}^0$. Prepared with this notations, we are ready to formulate the PFASST algorithm:

1. Go down to the coarse level:

- (a) Restrict the fine values $U_{[t_{l-1}, t_l]}^k$ to the coarse values $\tilde{U}_{[t_{l-1}, t_l]}^k$ and compute $\tilde{\mathbf{F}}_{[t_{l-1}, t_l]}^k$.
- (b) Compute FAS correction τ^k , using $\tilde{\mathbf{F}}_{[t_{l-1}, t_l]}^k$ and $\mathbf{F}_{[t_{l-1}, t_l]}^k$.
- (c) If $l > 0$, then receive the new initial value \tilde{U}_l^k from processor \mathbf{P}_{l-1} and compute $\tilde{\mathbf{F}}_{[t_{l-1}, t_l], 0}^k$, else use the initial value of the ODE.
- (d) Perform n_C SDC sweeps with values $\tilde{U}_{[t_{l-1}, t_l]}^k$, $\tilde{\mathbf{F}}_{[t_{l-1}, t_l]}^k$ and the FAS correction τ^k . This yields new values $\tilde{U}_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$ and $\tilde{\mathbf{F}}_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$.
- (e) Send $\tilde{U}_{[t_{l-1}, t_l], M_l}^{k+\frac{1}{2}}$ to processor \mathbf{P}_{l+1} if $l < N - 1$. This will be received as the new initial condition \tilde{U}_l^k for the solver on the coarse level.

2. Return to the fine level:

- (a) Interpolate the coarse correction $\delta^k = \tilde{U}_{[t_{l-1}, t_l]}^{k+\frac{1}{2}} - \tilde{U}_{[t_{l-1}, t_l]}^k$ and add to $U_{[t_{l-1}, t_l]}^k$, yielding $U_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$. Recompute $\mathbf{F}_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$.
- (b) If $l > 0$, then receive the new initial value \bar{U}_{l-1}^k from processor \mathbf{P}_{l-1} , else take the initial value of the ODE.
- (c) Interpolate coarse correction vector $\delta^k = \tilde{U}_{l-1}^{k+\frac{1}{2}} - \tilde{U}_{l-1}^k$ and add it to \bar{U}_l^k , yielding $\bar{U}_l^{k+\frac{1}{2}}$. Recompute $\mathbf{F}_{[t_{l-1}, t_l], 1}^{k+\frac{1}{2}}$.

3. Perform n_F fine SDC sweeps using the values $U_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$ and $\mathbf{F}_{[t_{l-1}, t_l]}^{k+\frac{1}{2}}$. This yields values $U_{[t_{l-1}, t_l]}^{k+1}$ and $\mathbf{F}_{[t_{l-1}, t_l]}^{k+1}$.

4. Send $U_{[t_{l-1}, t_l], M_l}^{k+1}$ to processor \mathbf{P}_{l+1} if $l < N - 1$. This will be used as initial value \bar{U}_{l+1}^{k+1} in the next iteration on the fine level.

This form of the PFASST algorithm is suitable for implementation, but rather not for the mathematical analysis. It is especially difficult to capture how the parts influence each other. To overcome this limitation, we now change the perspective: Instead of building the algorithm in a “vertical” way (MLSDC on each subinterval), we look at all intervals at once in a “horizontal” way, i.e., we analyze how the different components of PFASST act on the full time-domain $[t_0, T]$.

3. A MULTIGRID PERSPECTIVE

In this section, the perspective is shifted from solvers on one specific subinterval to the interaction of the solvers on the whole time domain $[t_0, T]$. We begin with stating the composite collocation problem.

Definition 2. Let the interval $[t_0, T]$ be decomposed as in (2) into L subintervals $[t_l, t_{l+1}]$. On each subinterval a collocation problem in the form of (4), denoted by $\mathbf{M}_{[t_l, t_{l+1}]}$, is posed. The collocation matrix on the whole time domain is then defined as

$$\mathbf{M}_{[t_0, T]} = \begin{pmatrix} \mathbf{M}_{[t_0, t_1]} & & & \\ -\mathbf{N} & \mathbf{M}_{[t_1, t_2]} & & \\ & \ddots & \ddots & \\ & & -\mathbf{N} & \mathbf{M}_{[t_{L-1}, T]} \end{pmatrix} \in \mathbb{R}^{NML}, \text{ with}$$

$$\mathbf{N} = \begin{pmatrix} 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \otimes \mathbf{I}_N \in \mathbb{R}^{NM}.$$

The operator \mathbf{N} handles how the new starting value for the upcoming interval is produced. Furthermore, stacking together

$$\mathbf{c}_{[t_l, t_{l+1}]} = \begin{cases} \mathbf{U}_0, & \text{for } l = 0 \\ \mathbf{0}, & \text{for } l > 0 \end{cases} \in \mathbb{R}^{NM},$$

form the right-hand side $\mathbf{c}_{[t_0, T]}$ for the **composite collocation problem**

$$\mathbf{M}_{[t_0, T]} \begin{pmatrix} \mathbf{U}_{[t_0, t_1]} \\ \mathbf{U}_{[t_1, t_2]} \\ \vdots \\ \mathbf{U}_{[t_{L-1}, T]} \end{pmatrix} = \begin{pmatrix} \mathbf{U}_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{c}_{[t_0, T]}. \quad (12)$$

Along with the definition, the block structure of our problem becomes evident. On the diagonal of the new collocation matrix, we find blocks of the size NM , each of them being associated with the subintervals $[t_l, t_{l+1}]$. The operators on the subdiagonal deal with the communication between two adjacent subintervals. When designing iterative solvers for the composite collocation problem, we also want to exploit this block structure. Therefore, the next two sections are dedicated to the block versions of an approximate Jacobi and a approximate Gauß-Seidel iteration and both will emerge from the interpretation of SDC as an approximate Gauß-Seidel iterative solver. Later on, both methods, if correctly interlaced, will yield PFASST.

3.1. Approximative Block Gauß-Seidel solver

The classical Gauß-Seidel solver is a splitting method, which incorporates the lower triangular part of the system matrix as preconditioner. Obviously this strategy is possible in principle for the composite collocation problem, as defined in Definition 2, but this would neglect the particular block structure of the problem. Therefore, we now construct a block version of the SDC iteration, following its description as an approximate Gauß-Seidel solver.

Assume we perform one SDC sweep on each subinterval via

$$\mathbf{U}_{[t_l, t_{l+1}]}^{k+1} = \mathbf{U}_{[t_l, t_{l+1}]}^k + \mathbf{P}_{[t_l, t_{l+1}]}^{-1} \left(\mathbf{c}_{[t_l, t_{l+1}]}^{k+1} - \mathbf{M}_{[t_l, t_{l+1}]} \mathbf{U}_{[t_l, t_{l+1}]}^k \right), \quad (13)$$

where $\mathbf{P}_{[t_l, t_{l+1}]}$ denotes the SDC preconditioner (6), and $\mathbf{c}_{[t_l, t_{l+1}]}^k$ is the right-hand side on the l -th subinterval in the k -th iteration. In order to pass the last value forward in time to the next subinterval, we can use the matrix \mathbf{N} . Therefore, the right-hand side of the collocation problem can be written as

$$\begin{aligned} \mathbf{c}_{[t_0, t_1]}^k &= [U_0, \dots, U_0], \quad \text{for } l = 0 \text{ and } k > 0 \\ \mathbf{c}_{[t_l, t_{l+1}]}^k &= [\bar{U}_l^k, \dots, \bar{U}_l^k] = \mathbf{N} \mathbf{U}_{[t_{l-1}, t_l]}^k \quad \text{for } l > 0 \text{ and } k > 1 \end{aligned} \quad (14)$$

For some initial iteration vector $\mathbf{U}_{[t_l, t_{l+1}]}^0$, stemming, e.g., from copying the initial value on each node of each subinterval (“spreading”), we can write this process compactly as single approximate Gauß-Seidel step over the whole time domain.

Lemma 2. *Let $\mathbf{M}_{[t_0, T]}$ be the matrix of a composite collocation problem. Using (13) on each subinterval and passing the results via (14), corresponds to*

$$\mathbf{U}_{[t_0, T]}^{k+1} = \mathbf{U}_{[t_0, T]}^k + \mathbf{P}_{[t_0, T]}^{-1} (\mathbf{c}_{[t_0, T]} - \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^k), \quad (15)$$

with

$$\mathbf{U}_{[t_0, T]}^k = \begin{pmatrix} \mathbf{U}_{[t_0, t_1]}^k \\ \mathbf{U}_{[t_1, t_2]}^k \\ \vdots \\ \mathbf{U}_{[t_{L-1}, T]}^k \end{pmatrix} \in \mathbb{R}^{NML}, \quad \mathbf{c}_{[t_0, T]} = \begin{pmatrix} \mathbf{U}_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{NML}$$

and

$$\mathbf{P}_{[t_0, T]} = \begin{pmatrix} \mathbf{P}_{[t_0, t_1]} & & & \\ -\mathbf{N} & \mathbf{P}_{[t_1, t_2]} & & \\ & \ddots & \ddots & \\ & & -\mathbf{N} & \mathbf{P}_{[t_{L-1}, T]} \end{pmatrix} \in \mathbb{R}^{NML \times NML}.$$

Proof

We multiply equation (13) with $\mathbf{P}_{[t_l, t_{l+1}]}$ from the left and equation (15) with $\mathbf{P}_{[t_0, T]}$ from the left. Comparing the resulting terms line by line reveals the equivalence. \square

This Gauss-Seidel-like iteration can be found in Fig. 1: Here, after each blue block which represent SDC sweeps on the coarse level, the values \bar{U}_l^k are passed forward in time, providing new initial values for the sweep on the next interval. Thus, the iteration on the coarse level can be identified with an approximate block Gauß-Seidel iteration for the composite collocation problem (12).

3.2. Block Jacobi-Solver

The communication, emerging from the use of the approximate Block Gauß-Seidel solver, is blocking. Each processor has to wait for its predecessor. Hence, this is a purely serial approach. A simple way to avoid the blocking communication is to use a approximate Block Jacobi solver, omitting the sub diagonal blocks responsible for the communication.

Assume we perform a step similar to (13), but we use the right-hand side

$$\begin{aligned} \mathbf{c}_{[t_0, t_1]}^k &= [U_0, \dots, U_0], \quad \text{for } l = 0 \text{ and } k > 0 \\ \mathbf{c}_{[t_l, t_{l+1}]}^k &= [\bar{U}_l^{k-1}, \dots, \bar{U}_l^{k-1}] = \mathbf{N} \mathbf{U}_{[t_{l-1}, t_l]}^{k-1} \quad \text{for } l > 0 \text{ and } k > 1 \end{aligned} \quad (16)$$

instead. This means that not the result of the current but of the previous iteration of the preceding interval is used. In the first iteration, the result of the prediction phase is used. Using the simple spreading prediction phase, this is easily achieved by choosing $\bar{U}_l^0 = U_0$.

Lemma 3. Let $\mathbf{M}_{[t_0, T]}$ be the matrix of a composite collocation problem. Then, using (13) on each subinterval and passing the results via (16), corresponds to

$$\mathbf{U}_{[t_0, T]}^{k+1} = \mathbf{U}_{[t_0, T]}^k + \hat{\mathbf{P}}_{[t_0, T]}^{-1} (\mathbf{c}_{[t_0, T]} - \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^k) \quad (17)$$

with

$$\hat{\mathbf{P}}_{[t_0, T]} = \begin{pmatrix} \mathbf{P}_{[t_0, t_1]} & & & \\ & \mathbf{P}_{[t_1, t_2]} & & \\ & & \ddots & \\ & & & \mathbf{P}_{[t_{L-1}, T]} \end{pmatrix},$$

as well as $\mathbf{U}_{[t_0, T]}^k$ and $\mathbf{c}_{[t_0, T]}$ defined as in Lemma 2.

Proof

Similar to the proof in Lemma 2 a block line-wise comparison yields the equivalence. Especially, the influence of the sub diagonal of $\mathbf{M}_{[t_0, T]}$ on the communication is revealed by a block line-wise view on (17):

$$\begin{aligned} \mathbf{U}_{[t_0, t_1]}^{k+1} &= \mathbf{U}_{[t_0, t_1]}^k + \mathbf{P}_{[t_0, t_1]}^{-1} (\mathbf{U}_0 - \mathbf{M}_{[t_0, t_1]} \mathbf{U}_{[t_0, t_1]}^k), & \text{for } l = 0 \\ \mathbf{U}_{[t_l, t_{l+1}]}^{k+1} &= \mathbf{U}_{[t_l, t_{l+1}]}^k + \mathbf{P}_{[t_l, t_{l+1}]}^{-1} (\mathbf{N} \mathbf{U}_{[t_{l-1}, t_l]}^k - \mathbf{M}_{[t_l, t_{l+1}]} \mathbf{U}_{[t_l, t_{l+1}]}^k), & \text{for } l > 0. \end{aligned}$$

The values $\mathbf{N} \mathbf{U}_{[t_{l-1}, t_l]}^k$ are equivalent to $\mathbf{1}_M \otimes \bar{\mathbf{U}}_l^{k-1}$. \square

It is evident that due to the block diagonal structure of $\hat{\mathbf{P}}_{[t_0, T]}$ one block Jacobi iteration may be performed concurrently on L computing units. This approach corresponds to the sweeps on the fine (red) blocks in Fig. 1: these sweeps can be performed in parallel, since they do not depend on the previous subinterval at the same iteration. Therefore, the iteration on the fine level can be identified with an approximate block Jacobi iteration for the composite collocation problem (12).

3.3. Assembling PFASST

Already in Section 2.3 multigrid elements were introduced to SDC to form MLSDC. The same ideas apply when we now interlace both iterative block solvers from above. In order to achieve more parallelism, we compute the approximate Gauß-Seidel iteration step on the coarse level and the approximate block Jacobi iteration step on the fine level, so that the more cost intensive work is done in parallel. As the following Theorem shows, it is now possible to write PFASST in the form of (8) and we are able to state a iteration matrix.

Theorem 1. Let \mathbf{T}_F^C and \mathbf{T}_C^F be block-wise defined transfer operators, which treat the subintervals independently from each other, let $\{\mathbf{P}_{[t_0, t_1]}, \dots, \mathbf{P}_{[t_{L-1}, T]}\}$ and $\{\tilde{\mathbf{P}}_{[t_0, t_1]}, \dots, \tilde{\mathbf{P}}_{[t_{L-1}, T]}\}$ be sets of preconditioner for the fine and coarse level, respectively, describing SDC sweeps on $[t_l, t_{l+1}]$ for $l \in \{0, \dots, L-1\}$ and $t_L = T$. Let $\mathbf{M}_{[t_0, T]}$ be the composite collocation matrix of Definition 2 and $\mathbf{N}, \tilde{\mathbf{N}}$ be the operations to compute the initial value for the following subinterval. Then the linear two-level version of PFASST can be written in matrix form as

$$\begin{aligned} \mathbf{U}_{[t_0, T]}^{k+\frac{1}{2}} &= \mathbf{U}_{[t_0, T]}^k + \mathbf{T}_C^F \tilde{\mathbf{P}}_{[t_0, T]}^{-1} \mathbf{T}_F^C (\mathbf{c}_{[t_0, T]} - \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^k) \\ \mathbf{U}_{[t_0, T]}^{k+1} &= \mathbf{U}_{[t_0, T]}^k + \hat{\mathbf{P}}_{[t_0, T]}^{-1} (\mathbf{c}_{[t_0, T]} - \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^{k+\frac{1}{2}}), \end{aligned} \quad (18)$$

with $\tilde{\mathbf{P}}_{[t_0, T]}$, as in Lemma 2, and $\hat{\mathbf{P}}_{[t_0, T]}$, as in Lemma 3. In addition, let $\mathbf{N}, \tilde{\mathbf{N}}$, such that $\tilde{\mathbf{N}} \mathbf{T}_F^C = \mathbf{T}_F^C \mathbf{N}$ and $\mathbf{c}_{[t_0, T]} = [\mathbf{U}^0, 0, \dots, 0]$ as well as $\tilde{\mathbf{c}}_{[t_0, T]} = \mathbf{T}_F^C \mathbf{c}_{[t_0, T]}$. Finally, following (7), the PFASST iteration matrix is given by

$$\mathbf{T}_{\text{PFASST}} = \left(\mathbf{I} - \hat{\mathbf{P}}_{[t_0, T]}^{-1} \mathbf{M}_{[t_0, T]} \right) \left(\mathbf{I} - \mathbf{T}_C^F \tilde{\mathbf{P}}_{[t_0, T]}^{-1} \mathbf{T}_F^C \mathbf{M}_{[t_0, T]} \right). \quad (19)$$

Proof

We compare systematically each step of PFASST with the sub-computations found in equation (18), which expands into

$$\tau_{[t_0, T]}^k = \tilde{\mathbf{M}}_{[t_0, T]} \mathbf{T}_F^C \mathbf{U}_{[t_0, T]}^k - \mathbf{T}_F^C \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^k \quad (20)$$

$$\tilde{\mathbf{U}}_{[t_0, T]}^{k+1} = \tilde{\mathbf{U}}_{[t_0, T]}^k + \tilde{\mathbf{P}}^{-1} \left(\tilde{\mathbf{c}}_{[t_0, T]} + \tau_{[t_0, T]}^k - \tilde{\mathbf{M}}_{[t_0, T]} \tilde{\mathbf{U}}_{[t_0, T]}^k \right) \quad (21)$$

$$\mathbf{U}_{[t_0, T]}^{k+\frac{1}{2}} = \mathbf{U}_{[t_0, T]}^k + \mathbf{T}_C^F \left(\tilde{\mathbf{U}}_{[t_0, T]}^{k+1} - \mathbf{T}_F^C \mathbf{U}_{[t_0, T]}^k \right) \quad (22)$$

$$\mathbf{U}_{[t_0, T]}^{k+1} = \mathbf{U}_{[t_0, T]}^{k+\frac{1}{2}} + \hat{\mathbf{P}}^{-1} \left(\mathbf{c}_{[t_0, T]} - \mathbf{M}_{[t_0, T]} \mathbf{U}_{[t_0, T]}^{k+\frac{1}{2}} \right). \quad (23)$$

From top to bottom, we have the computation of the FAS correction τ^k , the SDC sweep on the coarse level, coarse-grid correction, and the SDC sweep on the fine level. PFASST's communication between the subintervals has been already derived in Lemma 2 and Lemma 3. The evaluations of right-hand side in the form of \mathbf{F} and $\tilde{\mathbf{F}}$ are included in the matrix vector multiplication with $\mathbf{M}_{[t_0, T]}$ and $\tilde{\mathbf{M}}_{[t_0, T]}$, respectively.

The computation of the FAS correction $\tau_{[t_0, T]}^k$ as in (21) differs from the formula (11), which we derived for MLSDC, i.e. which is formed for each subinterval. The FAS correction vector of (21), has additional terms:

$$\mathbf{L} = \mathbf{T}_F^C \mathbf{N} - \tilde{\mathbf{N}} \mathbf{T}_F^C \quad (24)$$

with

$$\tau_{[t_0, T]} = \left(\tau_{[t_0, t_1]}, \tau_{[t_1, t_2]} + \mathbf{L} \mathbf{U}_{[t_0, t_1]}^k, \dots, \tau_{[t_{N-1}, T]} + \mathbf{L} \mathbf{U}_{[t_{N-1}, T]}^k \right)^T \quad (25)$$

However, by requirement we have $\mathbf{L} = \mathbf{0}$ and in Remark 1 we will investigate how this requirement is met. The iteration matrix is the result of simple algebraic manipulations. \square

In contrast to Lemma 1 for MLSDC, we now have an additional requirement.

Remark 1. Let $t_{i,j}$ be the j -th entry of the i -th row of \mathbf{T}_F^C . Due to the assumptions above, $\mathbf{L} = \mathbf{0}$ translate to

$$\begin{pmatrix} t_{\tilde{M},1} & \cdots & t_{\tilde{M},M-1} & t_{\tilde{M},M} \\ \vdots & & \vdots & \vdots \\ t_{\tilde{M},1} & \cdots & t_{\tilde{M},M-1} & t_{\tilde{M},M} \end{pmatrix} = \begin{pmatrix} 0 & \cdots & 0 & \sum_{j=1}^M t_{1,j} \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & \sum_{j=1}^M t_{\tilde{M},j} \end{pmatrix}.$$

Hence, we require that

$$t_{\tilde{M},j} = 0 \quad \forall j \in \{1, \dots, M-1\} \quad \text{and} \quad t_{\tilde{M},M} = \sum_{j=1}^M t_{i,j} \quad \forall i \in \{1, \dots, \tilde{M}\}.$$

If the restriction \mathbf{T}_F^C of a constant vector yields a constant vector with the same values but a smaller dimension, we infer that,

$$\sum_{j=1}^M t_{i,j} = 1 \quad \forall i \in \{1, \tilde{M}\}$$

and hence $t_{\tilde{M},M} = 1$. This requirement is met, when the restriction just projects the last node of the fine level onto the last node of the coarse level. It holds e.g. for the simple linear restriction or just injection, as long as the quadrature nodes $\tilde{\tau}_{\tilde{M}}$ and τ_M overlap for each subinterval.

The hierarchy of discretization on which PFASST is working and the exchange of information between those levels using FAS and coarse-grid correction obviously indicates a strong similarity to classical multigrid methods. This relation is in particular emphasized by the iteration matrix. Standard multigrid methods are typically described and analyzed by their iteration matrix \mathbf{T}_{MG} , which reads

$$\mathbf{T}_{\text{MG}}(\nu, \mu) = \left(\mathbf{I} - \mathbf{P}_{\text{post}}^{-1} \mathbf{M} \right)^{\nu} \left(\mathbf{I} - \mathbf{T}_C^F \tilde{\mathbf{M}}^{-1} \mathbf{T}_F^C \mathbf{M} \right) \left(\mathbf{I} - \mathbf{P}_{\text{pre}}^{-1} \mathbf{M} \right)^{\mu}, \quad (26)$$

for ν post- and μ pre-smoothing steps. The expression in the middle is the coarse grid correction. In a standard two-grid algorithm, the exact solution $\tilde{\mathbf{M}}^{-1}$ is used at the coarse level. In practice it is also legitimate to use the approximate solution in form of $\tilde{\mathbf{P}}^{-1}$. PFASST does exactly this. Under the conditions of Theorem 1, the comparison of (26) and (19) yields that PFASST can be readily interpreted as multigrid algorithm with one post-smoothing iteration and no pre-smoothing steps. We point out that this does not prove that PFASST actually behaves like a multigrid method in terms of convergence and robustness. In particular, properties like smoothing and approximation property are not necessarily satisfied and the analysis of the algorithm in this respect is left for future work. However, this does not prohibit an analysis based on the tools which are usually used for multigrid schemes.

4. LOCAL FOURIER ANALYSIS FOR PFASST

The most common tool for analysis and design of multigrid algorithms is the Local Fourier Analysis (LFA, see e.g. [17]). It simplifies the problem by making assumptions like periodic domains and constant coefficients. The goal of LFA is, in the rigorous case, the computation and usually the estimation of the spectral radius of the iteration matrix and its building blocks.

In this work we focus on two prototype problems, namely the diffusion and advection problem in one dimension, to show how PFASST can be analyzed in principle. We will use periodicity in space to stay rigorous in that dimension.

The usual approach to LFA is to define and work with Fourier symbols for each operator. These Fourier symbols represent the behavior of the operators on the grid functions

$$\varphi_{\theta}(x) = \exp(i\theta x/h), \quad x \in [0, 1], \quad \theta \in [-\pi, \pi], \quad (27)$$

for distinct frequencies θ . The observation, how the different grid functions are damped or changed on different grids and under different operations is a central point of LFA.

However, in our analysis we will make use of the matrix notation and henceforth avoid the use of explicit Fourier symbols, but rather perform a block diagonalization of the matrices of PFASST. The goal is the block-wise diagonalization of the iteration matrix of PFASST. Later on, each block will be associated with a discrete frequency. Therefore, we will be able to state which frequency is damped or changed to which extend.

Due to the periodicity in space, parts of the iteration matrix consists of circulant matrices. A circulant matrix is a special kind of Toeplitz matrix where each row vector is rotated one element to the right relative to the preceding row vector and denoted as

$$\mathbf{C} = \begin{pmatrix} c_0 & c_1 & \cdots & c_{\frac{N}{2}-1} \\ c_{\frac{N}{2}-1} & c_0 & & \\ & \ddots & \ddots & \\ c_1 & \cdots & c_{\frac{N}{2}-1} & c_0 \end{pmatrix}. \quad (28)$$

It has the eigenvalues λ_k and eigenvectors ψ_k for $k = 0, \dots, N-1$

$$\lambda_k = \sum_{j=0}^{N-1} c_j \exp\left(i \frac{2\pi}{N} k \cdot j\right) \quad \text{and} \quad (29)$$

$$\psi_k = \frac{1}{\sqrt{N}} \left[\exp\left(i \frac{2\pi}{N} k \cdot 0\right), \exp\left(i \frac{2\pi}{N} k \cdot 1\right), \dots, \exp\left(i \frac{2\pi}{N} k \cdot (N-1)\right) \right]^T.$$

This also means that with the transformation matrix Ψ , which is orthogonal and consists of the eigenvectors, it holds

$$\left(\Psi^T \mathbf{C} \Psi\right)_{j,j} = \lambda_j. \quad (30)$$

For two diagonalizable matrices \mathbf{A}, \mathbf{B} with the same eigenvector space it holds:

$$\begin{aligned} \Psi^T (\mathbf{A} + \mathbf{B}) \Psi &= \Psi^T \mathbf{A} \Psi + \Psi^T \mathbf{B} \Psi = \mathbf{D}^{(A)} + \mathbf{D}^{(B)}, \\ \Psi^T \mathbf{A} \mathbf{B} \Psi &= \Psi^T \mathbf{A} \Psi \Psi^T \mathbf{B} \Psi = \mathbf{D}^{(A)} \mathbf{D}^{(B)}, \\ \Psi^T \mathbf{A}^{-1} \Psi &= \left(\mathbf{D}^{(A)}\right)^{-1} \end{aligned} \quad (31)$$

Furthermore, for the Kronecker product we have $\mathcal{P}^{-1} \mathbf{A} \otimes \mathbf{B} \mathcal{P} = \mathbf{B} \otimes \mathbf{A}$, where \mathcal{P} is a suitable permutation matrix. Those rules will be used extensively by the following algebraic manipulations.

4.1. Transforming the PFASST iteration matrix

The PFASST algorithm has 3 layers it works on. The first layer is the spatial space, the second consists of the quadrature nodes, and the third is the temporal structure given by the subintervals. All layers are interweaved: we illustrate this by rewriting the system matrix $\mathbf{M}_{[t_0, T]}$ under the assumption that we have the same problem (i.e. the same discretization of the same operator) on each subinterval

$$\mathbf{M}_{[t_0, T]} = \mathbf{I}_L \otimes \mathbf{I}_M \otimes \mathbf{I}_N - \mathbf{I}_L \otimes \mathbf{Q} \otimes \mathbf{A} - \mathbf{E} \otimes \mathbf{N} \otimes \mathbf{I}_N, \quad (32)$$

where N is again the number of degrees of freedom in the spatial dimension, M the number of nodes per subinterval, and L the number of subintervals. Also, a new operator $\mathbf{E} \in \mathbb{R}^{L \times L}$ is introduced, which has ones on the first subdiagonal and zeros elsewhere. In each term the layers are separated by the Kronecker product, and through the summation of those parts we interweave them again. Our transformation aims at the layer, where each matrix is diagonalizable by Ψ .

We define a transformation matrix \mathcal{F} , which effects all layers, as

$$\mathcal{F} = \mathcal{P} \cdot (\mathbf{I}_L \otimes \mathbf{I}_N \otimes \Psi), \quad \mathcal{F}^{-1} = (\mathbf{I}_L \otimes \mathbf{I}_N \otimes \Psi^T) \cdot \mathcal{P}^{-1},$$

and therefore

$$\mathcal{F}^{-1} \mathbf{M}_{[t_0, T]} \mathcal{F} = \mathbf{I}_N \otimes (\mathbf{I}_L \otimes \mathbf{I}_M - \mathbf{E} \otimes \mathbf{N}) - \mathbf{D}^{(A)} \otimes \mathbf{I}_L \otimes \mathbf{Q}.$$

This yields diagonal matrices on the layer for the spatial dimension, so that we can write:

$$\begin{aligned} \mathcal{F}^{-1} \mathbf{M}_{[t_0, T]} \mathcal{F} &= \text{diag} \left(\mathbf{B}_1^{(M_{[t_0, T]})}, \dots, \mathbf{B}_N^{(M_{[t_0, T]})} \right) \\ \text{with } \mathbf{B}_j^{(M_{[t_0, T]})} &= \mathbf{I}_L \otimes \mathbf{I}_M - \mathbf{E} \otimes \mathbf{N} - \lambda_j \mathbf{I}_L \otimes \mathbf{Q}. \end{aligned}$$

We call the resulting blocks “time collocation blocks”, highlighting the dimension and components of the blocks. This idea was recently introduced in [37] in a different notation and is named “semi-algebraic mode analysis” (SAMA). The motivation behind SAMA is the large gap between the

theoretical analysis and the actual performance of multigrid methods for parabolic equations and fine-parallel methods. In [37] Friedhoff et al. demonstrated that SAMA enables accurate predictions of the short-term behavior and asymptotic convergence factors.

The transformation strategy above leads to a block structure for all matrices which emerge in the formulation of PFASST, in particular for the iteration matrix. Here, the interpolation and restriction matrices need special attention, though.

4.1.1. Transforming Interpolation and Restriction In this section we focus on interpolation and restriction operators, which are designed for two special isometric periodic grids with an even number of fine grid points. Between these two grids we define a special class of interpolation and restriction pairs.

Definition 3. Let $\mathbf{C} \in \mathbb{R}^{N \times N}$ be a circulant matrix, with the associated eigenvalues $\{\lambda_k\}_{k=1 \dots \frac{N}{2}}$, and let the fine grid X and coarse grid \tilde{X} be defined as

$$X = [x_1, \dots, x_N] \text{ and } \tilde{X} = [\tilde{x}_1, \dots, \tilde{x}_{N/2}], \text{ with } x_{2j-1} = \tilde{x}_j \text{ for all } j \in \left\{1, \dots, \frac{N}{2}\right\}.$$

Let $\mathcal{W}(\cdot, \cdot) : \mathbb{R}^{N \times N} \times \mathbb{R}^{N \times N} \mapsto \mathbb{R}^{2N \times N}$ be an “interweaving” operator, which stacks together the rows of two matrices subsequently, beginning with the first row of the first matrix, followed by the first row of the second matrix and finally ending with the last row of the second matrix. Then we define the class of circulant interweaved interpolation (“CI-interpolation”) operators as

$$\Pi = \left\{ \mathbf{T}_C^F : \exists \mathbf{C} \in \mathbb{R}^{N \times N} \text{ circulant and } \mathbf{C} \cdot \mathbf{1} = \mathbf{1}, \mathbf{T}_C^F = \mathcal{W}(\mathbf{I}_N, \mathbf{C}) \right\} \quad (33)$$

and the class of circulant interweaved restriction (“CI-restriction”) operators as

$$\Pi^T = \left\{ \mathbf{T}_F^C : c \left(\mathbf{T}_F^C \right)^T \in \Pi, c \in \mathbb{R} \right\}. \quad (34)$$

Due to the circulant nature of the interweaved matrices, we are able to state a transformation analytically.

Lemma 4. Let \mathbf{T}_C^F be a CI-interpolation and \mathbf{T}_F^C the associated CI-restriction operator, Ψ the transformation matrix for N grid points and Ψ_C the transformation matrix for $N/2$ grid points. Then it holds

$$\Psi^T \mathbf{T}_C^F \Psi_C = \begin{pmatrix} d_0 & & & \\ & \ddots & & \\ & & d_{N/2-1} & \\ \hat{d}_0 & & & \\ & \ddots & & \\ & & \hat{d}_{N/2-1} & \end{pmatrix} \quad (35)$$

and

$$\Psi_C^T \mathbf{T}_F^C \Psi = \frac{1}{2} \begin{pmatrix} d_0 & & \hat{d}_0 & \\ & \ddots & & \\ & & d_{N/2-1} & \\ & & & \hat{d}_{N/2-1} \end{pmatrix}. \quad (36)$$

The values on the diagonal depend solely on the circulant matrix \mathbf{C} and its eigenvalues $\lambda_k^{(C)}$ for $k \in \{0, N/2 - 1\}$. More precisely, we have

$$d_k = \frac{1 + \lambda_k^{(C)} \exp(-i \frac{2\pi}{N} k)}{\sqrt{2}} \quad \text{and} \quad \hat{d}_k = \frac{1 - \lambda_k^{(C)} \exp(-i \frac{2\pi}{N} k)}{\sqrt{2}}. \quad (37)$$

Proof

Using the properties of the interweaving operator we have

$$\mathbf{T}_C^F \cdot \Psi_C = \mathcal{W}(\mathbf{I}_{\frac{N}{2}}, \mathbf{C}_{\frac{N}{2}}) \Psi_C = \mathcal{W}(\Psi_C, \mathbf{C}_{\frac{N}{2}} \Psi_C).$$

Using the eigenvector eigenvalue relation (29) of the two circulant matrices \mathbf{C} and \mathbf{I} , see Section 4, for the computation of

$$[\mathbf{T}_C^F \cdot \Psi_C]_{-,k} = \sqrt{\frac{2}{N}} \begin{pmatrix} \exp(i4\pi/Nk \cdot 0) \\ \lambda_k \exp(i4\pi/Nk \cdot 0) \\ \vdots \\ \exp(i4\pi/Nk \cdot (N/2 - 1)) \\ \lambda_k \exp(i4\pi/Nk \cdot (N/2 - 1)) \end{pmatrix}.$$

A comparison to the immediate meaning of (35) demands

$$[\mathbf{T}_C^F \cdot \Psi_C]_{-,k} \stackrel{!}{=} \frac{d_k}{\sqrt{N}} \begin{pmatrix} \exp(i2\pi/Nk \cdot 0) \\ \vdots \\ \exp(i2\pi/Nk \cdot (N - 1)) \end{pmatrix} + \frac{\hat{d}_k}{\sqrt{N}} \begin{pmatrix} \exp(i2\pi/N(N/2 + k) \cdot 0) \\ \vdots \\ \exp(i2\pi/N(N/2 + k) \cdot (N - 1)) \end{pmatrix}.$$

Solving this system yields (37). □

Depending on the structure of \mathbf{C} , we are able to state further simplifications for d_k and \hat{d}_k , as we see in the following remark.

Remark 2. *For the special cases where \mathbf{C} has a symmetric stencil with*

$$c_l = \begin{cases} c_{\frac{N}{2}-l}, & \text{stencil length odd and } l \in 1, \dots, m \\ c_{\frac{N}{2}-(l+1)}, & \text{stencil length even and } l \in 0, \dots, m-1 \\ 0, & l > m \end{cases}$$

where $m \leq N/4$ for the even and $m \leq N/4 - 1/2$ for the odd case. Then, it holds for the odd case $d_k = d_{N/2-k}$ and $\hat{d}_k = \hat{d}_{N/2-k}$. In addition, for a CI-interpolation and -restriction operator with $\mathbf{C} \cdot \mathbf{1} = \mathbf{1}$ we have that $\lambda_0^{(C)} = 1$ and hence $d_0 = 0$ and $\hat{d}_0 = \sqrt{2}$.

We now use Lemma 4 to transform the coarse-grid correction. For the interpolation operator, we obtain diagonal entries $\{d_0, \hat{d}_0, \dots, d_{N/2-1}, \hat{d}_{N/2-1}\}$ and for the restriction operator the diagonal entries $\{f_0, \hat{f}_0, \dots, f_{N/2-1}, \hat{f}_{N/2-1}\}$. These entries may coincide if the same circulant matrix \mathbf{C} is used for the construction of both operators. Furthermore, we transform the inverse of the system matrix $\tilde{\mathbf{A}}^{-1}$ in the spatial dimension into a diagonal matrix consisting of the eigenvalues

$\{\tilde{\lambda}_0, \dots, \tilde{\lambda}_{N/2-1}\}$ of $\tilde{\mathbf{A}}^{-1}$. Then we obtain

$$\begin{aligned} \Psi^T \mathbf{T}_C^F \tilde{\mathbf{A}}^{-1} \mathbf{T}_F^C \Psi &= \Psi^T \mathbf{T}_C^F \Psi_C \Psi_C^T \tilde{\mathbf{A}}^{-1} \Psi_C \Psi_C^T \mathbf{T}_F^C \Psi \\ &= \frac{1}{2} \begin{pmatrix} d_0 & & & \\ & \ddots & & \\ & & d_{\frac{N}{2}-1} & \\ \hat{d}_0 & & & \\ & \ddots & & \\ & & \hat{d}_{\frac{N}{2}-1} & \end{pmatrix} \begin{pmatrix} \tilde{\lambda}_0 & & & \\ & \ddots & & \\ & & \tilde{\lambda}_{\frac{N}{2}-1} & \\ & & & \end{pmatrix} \begin{pmatrix} f_0 & & \hat{f}_0 & \\ & \ddots & & \\ & & f_{\frac{N}{2}-1} & \\ & & & \hat{f}_{\frac{N}{2}-1} \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} d_1 \tilde{\lambda}_0 f_0 & & \hat{d}_0 \tilde{\lambda}_0 f_0 & \\ & \ddots & & \ddots \\ & & d_{\frac{N}{2}-1} \tilde{\lambda}_{\frac{N}{2}-1} f_{\frac{N}{2}-1} & \\ d_0 \tilde{\lambda}_0 \hat{f}_0 & & \hat{d}_0 \tilde{\lambda}_0 \hat{f}_0 & \\ & \ddots & & \ddots \\ & & d_{\frac{N}{2}-1} \tilde{\lambda}_{\frac{N}{2}-1} \hat{f}_{\frac{N}{2}-1} & \\ & & & \hat{d}_{\frac{N}{2}-1} \tilde{\lambda}_{\frac{N}{2}-1} \hat{f}_{\frac{N}{2}-1} \end{pmatrix}. \end{aligned}$$

The values are now scattered over 3 diagonals. By using the appropriate permutation matrix we can gather them to new blocks:

$$\mathcal{P}^{-1} \Psi^T \mathbf{T}_C^F \tilde{\mathbf{A}}^{-1} \mathbf{T}_F^C \Psi \mathcal{P} = \text{diag} \left(\mathbf{B}_0, \dots, \mathbf{B}_{\frac{N}{2}-1} \right), \quad (38)$$

$$\text{where } \mathbf{B}_l = \begin{pmatrix} d_l \tilde{\lambda}_l f_l & \hat{d}_l \tilde{\lambda}_l f_l \\ d_l \tilde{\lambda}_l \hat{f}_l & \hat{d}_l \tilde{\lambda}_l \hat{f}_l \end{pmatrix} \in \mathbb{R}^{2 \times 2}. \quad (39)$$

In this structure we find the classical mode-mixing property of interpolation and restriction operators. This well-known property of standard multigrid iterations interweaves pairs of one low and one high frequency, the “harmonics”.

4.1.2. Transforming the full iteration matrix The iteration matrix of PFASST can now be transformed into a block matrix with $N/2$ blocks of the size $M \cdot L$. Each block is associated with a harmonic of the spatial problem and therefore with one high and one low frequency. In contrast, the smoother alone is decomposed into N blocks, which may be associated with only one single frequency. This is summarised in the following theorem.

Theorem 2. *Let us have a iteration matrix in the form of (19) with*

$$\mathbf{T} = \left(\mathbf{I} - \mathbf{P}^{-1} \mathbf{M} \right) \left(\mathbf{I} - \mathbf{T}_C^F \tilde{\mathbf{P}}^{-1} \mathbf{T}_C^M \right),$$

where \mathbf{M} is the collocation matrix, $\mathbf{T}_F^C, \mathbf{T}_C^F$ are two circulant interweaved transfer operators and $\mathbf{P}, \tilde{\mathbf{P}}$ are two preconditioner with a matrix in the spatial layer, which is diagonalisable and has the same eigenvector space as the spatial system matrix \mathbf{A} . Then there exists a transformation \mathcal{F} so that

$$\mathcal{F}^{-1} \mathbf{T} \mathcal{F} = \text{diag} \left(\mathcal{B}_0^{(S)} \mathcal{B}_0^{(CGC)}, \dots, \mathcal{B}_{\frac{N}{2}-1}^{(S)} \mathcal{B}_{\frac{N}{2}-1}^{(CGC)} \right) \in \mathbb{R}^{LMN \times LMN}, \text{ with} \quad (40)$$

$$\mathcal{B}_k^{(S)} = \begin{pmatrix} \mathbf{I} - \left(\mathbf{B}_k^{(P)} \right)^{-1} \mathbf{B}_k^{(M)} & \\ & \mathbf{I} - \left(\mathbf{B}_{\frac{N}{2}+k}^{(P)} \right)^{-1} \mathbf{B}_{\frac{N}{2}+k}^{(M)} \end{pmatrix} \in \mathbb{R}^{2LM \times 2LM} \quad (41)$$

$$\mathcal{B}_k^{(CGC)} = \begin{pmatrix} \mathbf{I} - f_k d_k \left(\mathbf{B}_k^{(\tilde{P})} \right)^{-1} \mathbf{B}_k^{(M)} & -\hat{f}_k d_k \left(\mathbf{B}_k^{(\tilde{P})} \right)^{-1} \mathbf{B}_{\frac{N}{2}+k}^{(M)} \\ -\hat{d}_k f_k \left(\mathbf{B}_k^{(\tilde{P})} \right)^{-1} \mathbf{B}_k^{(M)} & \mathbf{I} - \hat{f}_k \hat{d}_k \left(\mathbf{B}_k^{(\tilde{P})} \right)^{-1} \mathbf{B}_{\frac{N}{2}+k}^{(M)} \end{pmatrix} \in \mathbb{R}^{2LM \times 2LM}, \quad (42)$$

with matrices $B_k^{(P)}, B_k^{(M)} \in \mathbb{R}^{LM \times LM}$ for $k = 0 \dots N-1$ and $B_k^{(\tilde{P})} \in \mathbb{R}^{LM \times LM}$ for $k = 0 \dots \frac{N}{2} - 1$, solely depending on the eigenvalues of \mathbf{A} and $\tilde{\mathbf{A}}$. Where

$$\begin{aligned}\Psi^T \mathbf{P} \Psi &= \text{diag} \left(\mathbf{B}_0^{(P)}, \dots, \mathbf{B}_{N-1}^{(P)} \right), \text{ with } \mathbf{B}_j^{(P)} = \mathbf{I}_L \otimes \mathbf{I}_M - \mathbf{E} \otimes \mathbf{N} - \lambda_j^{(A)} \mathbf{I}_L \otimes \mathbf{Q}_\Delta, \\ \Psi^T \mathbf{M} \Psi &= \text{diag} \left(\mathbf{B}_0^{(M)}, \dots, \mathbf{B}_{N-1}^{(M)} \right), \text{ with } \mathbf{B}_j^{(M)} = \mathbf{I}_L \otimes \mathbf{I}_M - \mathbf{E} \otimes \mathbf{N} - \lambda_j^{(A)} \mathbf{I}_L \otimes \mathbf{Q}, \\ \Psi^T \tilde{\mathbf{P}} \Psi &= \text{diag} \left(\mathbf{B}_0^{(\tilde{P})}, \dots, \mathbf{B}_{\frac{N}{2}-1}^{(\tilde{P})} \right), \text{ with } \mathbf{B}_j^{(\tilde{P})} = \mathbf{I}_L \otimes \mathbf{I}_M - \mathbf{E} \otimes \mathbf{N} - \lambda_j^{(\tilde{A})} \mathbf{I}_L \otimes \mathbf{Q}_\Delta.\end{aligned}\tag{43}$$

We call $\mathbf{B}_j^{(M)}$, $\mathbf{B}_j^{(P)}$ and $\mathbf{B}_j^{(\tilde{P})}$ basic blocks. The matrix $\mathbf{Q}_\Delta \in \mathbb{R}^{M \times M}$ is a lower triangular matrix approximating \mathbf{Q} , see Section 2.2.

Proof

The proof consists of straightforward computations. The matrices \mathbf{P} , \mathbf{M} and $\tilde{\mathbf{P}}$ have 3 layers, separated by Kronecker products like in (32). Applying the transformation in the spatial dimension leads to the basic blocks (43). Similar to (39), we choose the adequate permutation matrices on the layers of subintervals and quadrature nodes, to get the blocks of harmonics. Also, each block of the post smoother is associated with a mode, hence we stack harmonic pairs together to $\mathcal{B}_k^{(S)}$ in order to match them with the blocks of the coarse grid correction $\mathcal{B}_k^{(CGC)}$, performed by the same permutation matrix. \square

This theorem makes it possible, at least semi algebraically, to analyze the convergence properties of PFASST by computing the spectral radius of each block $\mathcal{B}_k^{(S)} \mathcal{B}_k^{(CGC)}$. Until this point the choice of the particular problem and the operators yields a rigorous transformation. Hence, the blocks and the full iteration matrix of PFASST have exactly the same eigenvalues. This translates to computing $N/2$ eigenvalues of matrices of the size $2ML \times 2ML$. As we can see in (43), the basic blocks consists of \mathbf{I}_L and \mathbf{E} on the first layer. However, it is not directly possible to apply the transformation strategy presented above to this layer. For an empirical study like LFA, though, only estimates of the spectral radii are needed. This is mainly due to the fact, that even the exact spectral radius does not reflect the direct numerical behavior of the method exactly, but rather asymptotically. In the following section we therefore give up the rigorousness of the transformation in order to find a decomposition of the basic blocks into L blocks of the size $2M \times 2M$.

4.2. Assuming periodicity in time

To enable the further decomposition of the basic blocks, we exchange in the matrix formulation

$$\mathbf{E} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & & \\ \vdots & & \ddots & \\ 0 & & & 1 & 0 \end{pmatrix} \quad \text{with} \quad \hat{\mathbf{E}} = \begin{pmatrix} 0 & 0 & \cdots & 1 \\ 1 & 0 & & \\ \vdots & & \ddots & \\ 0 & & & 1 & 0 \end{pmatrix},$$

which introduces time periodicity to the problem and makes the matrix circulant. Hence, it becomes easy to transform

$$\left[\Psi^{-1} \hat{\mathbf{E}} \Psi \right]_{j,j} = \exp \left(-i2\pi \frac{j}{L} \right),$$

which makes the basic blocks $\mathbf{B}_j^{(M)}$, $\mathbf{B}_j^{(P)}$ and $\mathbf{B}_j^{(\tilde{P})}$ further decomposable into NL or $NL/2$ blocks of the size $M \times M$ or $2M \times 2M$, respectively. This leads directly to the following Theorem that can be proved using straightforward computations similar to the ones used before.

Theorem 3. Let us have the identical requirements as in Theorem 2, except the use of $\hat{\mathbf{E}}$ instead of \mathbf{E} . Then there exists a transformation \mathcal{F} such that

$$\hat{\mathcal{F}}^{-1} \mathbf{T} \hat{\mathcal{F}} = \text{diag} \left(\mathcal{B}_{0,0}^{(S)} \cdot \mathcal{B}_{0,0}^{(CGC)}, \mathcal{B}_{0,1}^{(S)} \cdot \mathcal{B}_{0,1}^{(CGC)}, \dots, \mathcal{B}_{\frac{N}{2}-1,L-1}^{(S)} \cdot \mathcal{B}_{\frac{N}{2}-1,L-1}^{(CGC)} \right), \quad (44)$$

$$\text{with } \mathcal{B}_{k,j}^{(S)} = \begin{pmatrix} \mathbf{I} - \left(\mathbf{B}_{k,j}^{(P)} \right)^{-1} \mathbf{B}_{k,j}^{(M)} & \\ & \mathbf{I} - \left(\mathbf{B}_{\frac{N}{2}+k,j}^{(P)} \right)^{-1} \mathbf{B}_{\frac{N}{2}+k,j}^{(M)} \end{pmatrix} \in \mathbb{R}^{2M \times 2M} \text{ and} \quad (45)$$

$$\mathcal{B}_{k,j}^{(CGC)} = \begin{pmatrix} \mathbf{I} - \hat{f}_k d_k \left(\mathbf{B}_{k,j}^{(\tilde{P})} \right)^{-1} \mathbf{B}_{k,j}^{(M)} & -\hat{f}_k d_k \left(\mathbf{B}_{k,j}^{(\tilde{P})} \right)^{-1} \mathbf{B}_{N/2+k,j}^{(M)} \\ -\hat{d}_k f_k \left(\mathbf{B}_{k,j}^{(\tilde{P})} \right)^{-1} \mathbf{B}_{k,j}^{(M)} & \mathbf{I} - \hat{f}_k \hat{d}_k \left(\mathbf{B}_{k,j}^{(\tilde{P})} \right)^{-1} \mathbf{B}_{N/2+k,j}^{(M)} \end{pmatrix} \in \mathbb{R}^{2M \times 2M}, \quad (46)$$

with matrices $\mathbf{B}_{k,j}^{(P)}, \mathbf{B}_{k,j}^{(M)} \in \mathbb{R}^{M \times M}$ for $k = 0 \dots N-1, j = 0 \dots L-1$ and $\mathbf{B}_{k,j}^{(\tilde{P})} \in \mathbb{R}^{M \times M}$ for $k = 0 \dots \frac{N}{2}-1, j = 0 \dots L-1$, solely depending on the eigenvalues of \mathbf{A} and $\tilde{\mathbf{A}}$, with

$$\begin{aligned} \Psi^T \mathbf{P} \Psi &= \text{diag} \left(\mathbf{B}_{0,0}^{(P)}, \dots, \mathbf{B}_{N-1,L-1}^{(P)} \right), \text{ with } \mathbf{B}_{k,j}^{(P)} = \mathbf{I} - \lambda_k^{(A)} \Delta t \mathbf{Q}_\Delta + \exp \left(-i2\pi \frac{j}{L} \right) \mathbf{N}, \\ \Psi^T \mathbf{M} \Psi &= \text{diag} \left(\mathbf{B}_{0,0}^{(M)}, \dots, \mathbf{B}_{N-1,L-1}^{(M)} \right), \text{ with } \mathbf{B}_{k,j}^{(M)} = \mathbf{I} - \lambda_k^{(A)} \Delta t \mathbf{Q} + \exp \left(-i2\pi \frac{j}{L} \right) \mathbf{N}, \\ \Psi^T \tilde{\mathbf{P}} \Psi &= \text{diag} \left(\mathbf{B}_{0,0}^{(\tilde{P})}, \dots, \mathbf{B}_{\frac{N}{2}-1,L-1}^{(\tilde{P})} \right), \text{ with } \mathbf{B}_{k,j}^{(\tilde{P})} = \mathbf{I} - \lambda_k^{(\tilde{A})} \Delta t \mathbf{Q}_\Delta + \exp \left(-i2\pi \frac{j}{L} \right) \mathbf{N}. \end{aligned} \quad (47)$$

We denote those blocks as “collocation blocks” in contrast to the time-collocation blocks of Theorem 2.

This leaves us with $NL/2$ blocks of the size $2M \times 2M$. We identify the matrices \mathbf{Q} and \mathbf{Q}_Δ as the atomic part of the whole matrix formulation. Further decompositions may only be performed if a decomposition of \mathbf{Q} is found. In the case of a $\mathbf{Q} \in \mathbb{R}^{1 \times 1}$, the time stepping part reduces to e.g. an implicit Euler. In this case no eigenvalue computations are necessary any more and the Fourier symbols are easily derived from the basic collocation blocks.

Remark 3. With the assumption of periodicity in time we loose the initial value, which means that if $u(t, x)$ is a solution of the problem then $u(t, x) + c$ is also a solution for any $c \in \mathbb{R}$. Hence, the inverses of $\mathbf{B}_{k,0}^{(P)}$ and $\mathbf{B}_{k,0}^{(\tilde{P})}$ do not exist and neither do the inverses of iteration matrix blocks $\mathcal{B}_{k,0}^{(T)} = \mathcal{B}_{k,0}^{(S)} \cdot \mathcal{B}_{k,0}^{(CGC)}$ exist. Our remedy for this problem is to set $\mathcal{B}_{k,0}^{(T)}$ to $\mathbf{0}$. This blocks belong to constant modes and we assume that there are no constant error modes which have to be damped.

Based on this transformation of the iteration matrix, we are now able to investigate the behavior of PFASST for two standard model problems in the following section.

5. NUMERICAL EXPERIMENTS

In this section we show how the convergence properties of PFASST may be examined along the lines of two examples, namely the diffusion and the advection problem. Within this paper though, a full analysis of the influence all the parameters like $N, L, M, \Delta t$, the choice of the quadrature rule or the PDE parameters is not possible. Therefore, the experiments presented here do not aim for a complete analysis, they should rather be viewed as a recipe to analyze PFASST for a certain class of problems, defined by the requirements we posed for the theoretical results above.

All computations are performed with $N = 128$ degrees of freedom in space. For matrices and vectors, the infinity norm is used. For the advection problem, we will use the SDC algorithm with

the LU-based preconditioner \mathbf{Q}_Δ as in [33], while for the diffusion problem \mathbf{Q}_Δ is the standard implicit Euler method. For all experiments we use $M = 5$ Gauß-Radau nodes on each of the $L = 4$ subintervals of the length $dt = 0.1$. Hence, we have $T = 0.4$. The interpolation is constructed such that polynomials up to order 6 are interpolated exactly and the restriction is gained from an interpolation operator which interpolates polynomials up to the order of 2.

Our main goal will be the estimation of the error by using the block form of the iteration matrix of PFASST. With blocks \mathbf{B}_k , the computation of the norm of a matrix \mathbf{H} reduces to

$$\|\mathbf{H}\|^2 = \sup_{\mathbf{x}_k \neq 0} \frac{\sum_{k=1}^m \|\mathbf{B}_k \mathbf{x}_k\|^2}{\sum_{k=1}^m \|\mathbf{x}_k\|^2} = \max_k \sup_{\mathbf{x}_k \neq 0} \frac{\|\mathbf{B}_k \mathbf{x}_k\|^2}{\|\mathbf{x}_k\|^2} = \max_k \|\mathbf{B}_k\|^2, \quad (48)$$

see [17] for a proof. The same holds for the computation of the spectral radii. In addition, the effort of computing the eigenvalues of $N/2$ time-collocation blocks of the size $2LM \times 2LM$ is obviously less than for a $MLN \times MLN$ matrix. With the assumption in Section 4.2 it even reduces to the computation of $NL/2$ collocation blocks of the size $2M \times 2M$.

For both cases (time collocation and collocation blocks) we consider the following strategies for the estimation of the error vector e^κ of the κ iteration

1. use the spectral radius $\rho(\mathbf{T})$ of the iteration matrix
2. use the norm of the iteration matrix $\|\mathbf{T}\|$
3. use the norm of the κ -th potency of the iteration matrix $\|\mathbf{T}^\kappa\|$
4. apply κ -th times the iteration matrix to the known error vector

The first strategy is based on the inequality for consistent matrix norms $\|\cdot\|$ and each $\kappa \in \mathbb{N}$

$$\rho(A) \leq \|A^\kappa\|^{\frac{1}{\kappa}}, \quad (49)$$

see [38]. Strategies 2 and 3 rely on the inequality

$$\|e^\kappa\| = \|\mathbf{T}^\kappa e^0\| \leq \|\mathbf{T}^\kappa\| \|e^0\| \leq \|\mathbf{T}\|^\kappa \|e^0\|. \quad (50)$$

Note that the iteration matrix is separated from the initial error vector e^0 , and therefore an a priori estimation of the relative error reduction is possible for this strategies. In contrast, the error vector e^0 , i.e. the analytical solution has to be known for strategy 4, making it an a posteriori strategy. If time collocation blocks are used, the computation following strategy 4 yields the analytically correct error for each iteration. Using collocation blocks, this approach just provides another estimate.

5.1. Diffusion problem

The elliptic Poisson problem is often used in the multigrid literature to demonstrate the basic ideas of multigrid, see e.g. [17]. Hence, the time-dependent, parabolic version of it, i.e. the classical heat equation, is a canonical candidate for the analysis of a multigrid-like time integration method like PFASST.

The problem in one spatial dimension is given by

$$\begin{aligned} u_t &= \nu \Delta u, & x &\in [0, 1] \text{ and } t \in [0, T] \\ u(x, 0) &= u_0(x), & u(0, t) &= u(1, t), \quad t \in [0, T] \end{aligned} \quad (51)$$

for a time $T > 0$ and the diffusion coefficient $\nu > 0$. Using second-order finite differences on a isometric grid we get a simple discretization in the spatial dimension with

$$X = [x_1, \dots, x_N], \text{ with } x_j = \frac{j-1}{N} \text{ and } \Delta x = \frac{1}{N} \quad (52)$$

which leads to a system of linear ODEs

$$U_t(t) = \mathbf{A} U(t), \quad t \in [0, T] \text{ and } U(0) = [u(x_1, 0), \dots, u(x_N, 0)],$$

$$\text{with } \mathbf{A} = \frac{\nu}{(\Delta x)^2} \begin{pmatrix} 2 & -1 & 0 & \cdots & -1 \\ -1 & 2 & -1 & & \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -1 & 2 & -1 \\ -1 & 0 & \cdots & -1 & 2 \end{pmatrix} \in \mathbb{R}^{N \times N}. \quad (53)$$

Because the matrix \mathbf{A} is circulant, the spectral decomposition in eigenvalues and eigenvectors is easily computed. For the eigenvalues λ_k and normal eigenvectors ψ_k , $k \in \{0, \dots, N-1\}$, we have

$$\lambda_k = \frac{4\nu}{\Delta x^2} \sin^2\left(\frac{k\pi}{N}\right) \quad \text{and} \quad \psi_k = \frac{1}{\sqrt{N}} \left[\exp\left(i\frac{2\pi}{N}k \cdot 0\right), \dots, \exp\left(i\frac{2\pi}{N}k \cdot (N-1)\right) \right]^T. \quad (54)$$

5.1.1. The error vector For initial values given by the function

$$u_0(x) = \sin(2\pi x k), \quad k \in \{1, \dots, N-1\}, \quad x \in [0, 1], \quad (55)$$

we know that solution to our PDE with periodic boundary conditions is given by

$$u(t, x) = \exp\left(-\nu(2\pi k)^2 t\right) \sin(2\pi k x).$$

Usually the PFASST algorithms starts with an vector where the initial value is spread on each node, i.e. we have the initial error vector

$$\mathbf{e}^0 = \begin{pmatrix} 1 - \exp(-\nu(2\pi k)^2 t_0 + \tau_1) \\ \vdots \\ 1 - \exp(-\nu(2\pi k)^2 T) \end{pmatrix} \otimes \begin{pmatrix} \sin(2\pi k x_1) \\ \vdots \\ \sin(2\pi k x_N) \end{pmatrix}.$$

With the iteration matrix we compute the succeeding error vector for PFASST as

$$\mathbf{T} \mathbf{e}^\kappa = \mathbf{e}^{\kappa+1}.$$

Like the iteration matrix, the error vector \mathbf{e}^k of the k -th iteration itself can be transformed and decomposed into parts belonging to a certain mode and associated with the TC-block of the iteration matrix. We can write

$$\mathcal{F}^{-1} \mathbf{T} \mathcal{F} \mathcal{F}^{-1} \mathbf{e}^\kappa = \mathcal{F}^{-1} \mathbf{e}^{\kappa+1}.$$

The transformed error is thus $\hat{\mathbf{e}}^\kappa = \mathcal{F}^{-1} \mathbf{e}^\kappa$, following precisely the transformation procedure described in Section 4.1. The initial value function $u_0(x)$ decomposes into two modes, which are represented by spatial Fourier space functions

$$\begin{aligned} \sin(2\pi k x_j) &= \frac{1}{2i} \exp\left(i\frac{2\pi}{N}k x_j\right) - \frac{1}{2i} \exp\left(-i\frac{2\pi}{N}k x_j\right) \\ &= \frac{\sqrt{N}}{2i} [\psi_k]_j - \frac{\sqrt{N}}{2i} [\psi_{N-k}]_j \end{aligned}$$

and belong to two different harmonics. This reduces our analysis to the blocks belonging to these certain harmonics, which are $\mathcal{B}_k^{(T)}, \mathcal{B}_{\frac{N}{2}-k}^{(T)}$ of sizes $2LM$ for $k = 0, \dots, N/2 - 1$ in the case time-collocation blocks are considered and $\mathcal{B}_{k,j}^{(T)}, \mathcal{B}_{\frac{N}{2}-k,j}^{(IT)}$ of sizes $2M$ for $k = 0, \dots, N/2 - 1$ and $j = 0, \dots, L - 1$ if collocation blocks are considered.

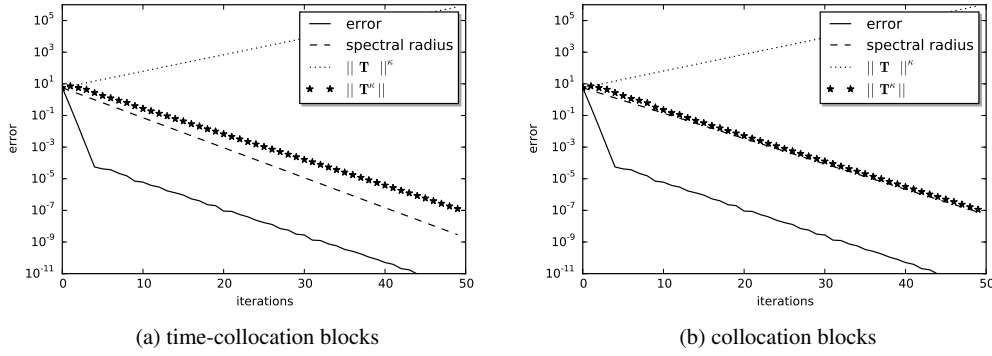


Figure 2. The errors estimates from the strategies 1 to 3, compared to the actual error plotted against the number of iterations of PFASST, for the diffusion problem with an initial value function of $\sin(2\pi 8x)$.

5.1.2. Error prediction We choose ν so that $\mu = 10$. In Figure 2, observing the solid line of the actual error measured during the iterations, we first see a short-term convergence behavior until roughly 10^{-4} which is then followed by a much slower, long-term convergence phase. We see that the use of the norm of the k -th potency of the iteration matrix \mathbf{T} (strategy 3) is well-suited to capture the long-term convergence behavior. This is of course also true for strategy 1, using the spectral radius of the iteration matrix. Similar plots for various initial value functions $\sin(2\pi kx)$ for $k \in \{1, \dots, N/2 - 1\}$, were inspected and showed the same behavior for the long-term convergence. In particular, there is no significant difference between time-collocation and collocation blocks. However, the norm of the iteration matrix is greater than 1 for most cases, as a survey over different $\mu \in (0.01, 100)$ and $L \in \{2, \dots, 50\}$ showed. This renders strategy 2 useless for most of the cases we considered so far.

The short-term convergence on the other hand is not captured by the first 3 strategies. In contrast, strategy 4 does this very well, as we see in Figure 3. We also see that the short-term convergence is faster for initial values with a small wave number k and that the long-term convergence speed is almost independent from the initial value. Our interpretation is that PFASST is more efficient in reducing the low frequency error modes in space. After the first convergence phase, the error consists of a mixture of modes, which is reduced by PFASST likewise, independently from the initial value frequency.

Here we actually see a difference between the different types of blocks: For the error prediction of the first iterations, using strategy 4 with collocation blocks is not as accurate as using time collocation blocks. In contrast, no differences in the quality of the error prediction are notable in the long-term convergence phase, again.

5.2. Advection problem

The second prototype problem is the 1D advection equation, given by

$$\begin{aligned} u_t &= cu_x, \quad x \in [0, 1] \text{ and } t \in [0, T] \\ u(x, 0) &= u_0(x), \quad u(0, t) = u(1, t) \quad t \in [0, T], \end{aligned} \quad (56)$$

with advection coefficient $c > 0$. The discretization is done in the same manner as (53), but we use an upwind difference stencil of the order 3, instead of a central difference stencil. This yields again a circulant matrix \mathbf{F}_D , with eigenvalues and eigenvectors according to (29). For the numerical experiments we use advection speed $c = 4.88 \cdot 10^{-3}$, resulting in a CFL number of $62.5 \cdot 10^{-3}$. The discretization in space and time is similar to the discretization in the previous section.

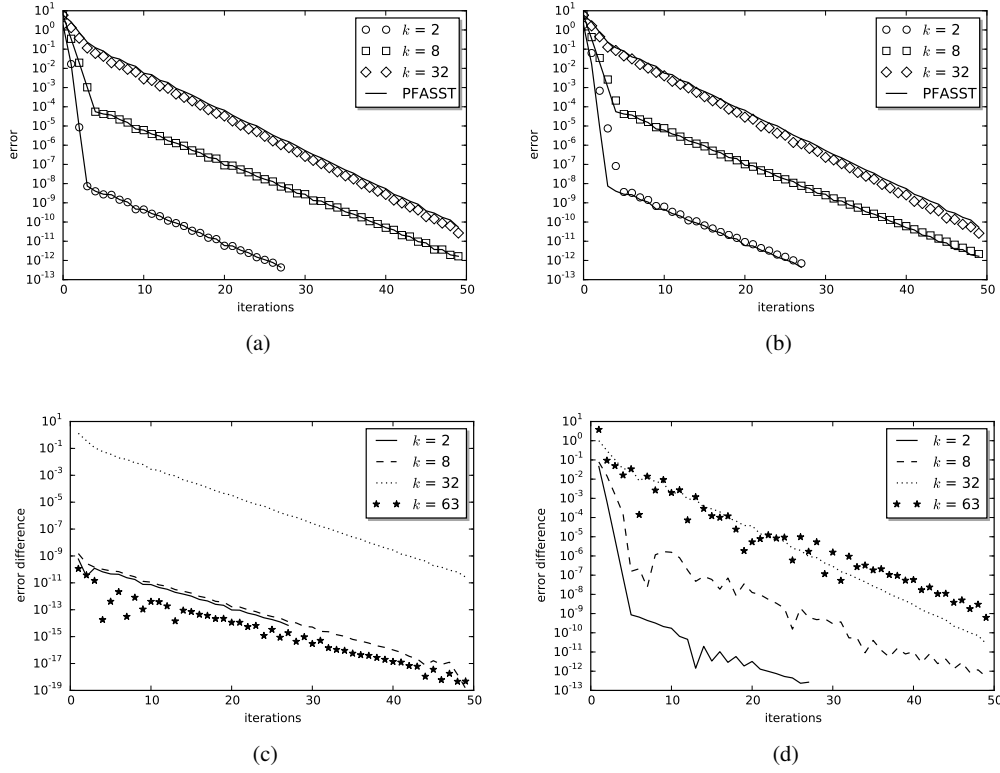


Figure 3. Strategy 4 for various k , using time collocation blocks on the left and collocation blocks on the right for the diffusion problem. In the second row the difference between the error and the error prediction is plotted.

5.2.1. *The error vector* For a initial value function u_0 the solution reads

$$u(t, x) = u_0(x - ct).$$

We use again the initial values given by (55). The initial values are spread on each node, this yields the initial error vector

$$\mathbf{e}^0 = (\mathbf{e}_1^0, \dots, \mathbf{e}_L^0)^T$$

with

$$\begin{aligned} \mathbf{e}_n^0 &= (\mathbf{e}_{n,1}^0, \dots, \mathbf{e}_{n,M}^0)^T \\ \mathbf{e}_{j,m}^0 &= (u_0(x_1) - u_0(x_1 - c(t_{j-1} + \tau_m)), \dots, u_0(x_N) - u_0(x_N - c(t_{j-1} + \tau_m))) \end{aligned}$$

When the class of initial value (55) is used, the initial values can be decomposed again into two modes, belonging to different harmonics. The analysis is thus again reduced to certain harmonic blocks $\mathcal{B}_k^{(T)}$, $\mathcal{B}_{\frac{N}{2}-k}^{(T)}$ and $\mathcal{B}_{k,j}^{(T)}$, $\mathcal{B}_{\frac{N}{2}-k,j}^{(T)}$, respectively. Note, that this computation of the error vector works even if only a numerical solution to the problem is given.

5.2.2. *Error prediction* For the advection problem the four strategies yield significantly different results than for the diffusion problem. In Fig. 4 we now observe three phases of convergence: two rapid phases at the beginning and at the end and one almost stagnating phase in the middle. We observed these phases for all initial wave numbers κ , with the peculiarity of a decreasing, almost vanishing first phase for increasing κ . Regarding the different strategies, we see that only the spectral

radii (strategy 1) is able to capture the first phase, while the norm of the powers of the iteration matrix (strategy 3) captures the last phase.

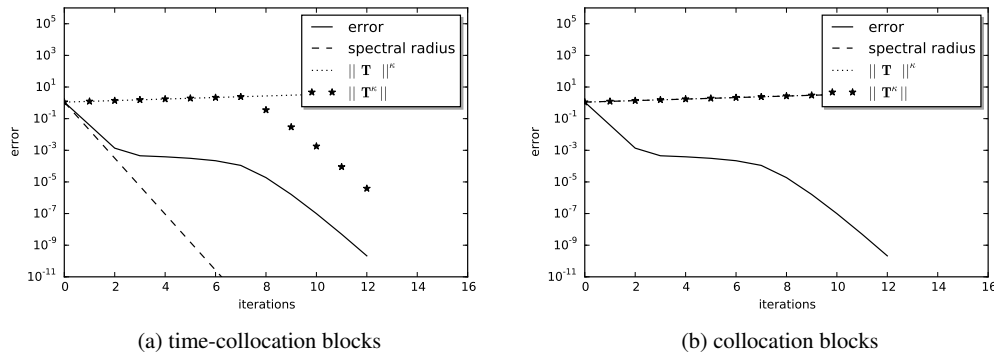


Figure 4. The errors estimates from the strategies 1 to 3, compared to the actual error plotted against the number of iterations of PFASST. Advection problem with an initial value function of $u_0(x) = \sin(2\pi 8x)$.

Again strategy 4 is successful in exactly predicting the error, when time collocation blocks are used. On the other hand, for the advection equation the use of collocation blocks only serve as an assessment for initial values with high k , and then only for the first phase. Obviously, the assumption of periodicity in time is not valid for advection-dominated problems. Thus, time collocation blocks should be considered in this case.

6. CONCLUSION AND OUTLOOK

In this paper we decomposed the PFASST algorithm into its atomic parts. Using analogies to classical iterative methods like Gauß-Seidel and Jacobi, we described PFASST for two levels and linear problems as a combination of a highly parallel, approximative block Jacobi solver on the fine level and a serial, approximative block Gauß-Seidel solver on the coarse level. With this we could show that for linear problems PFASST is a multigrid algorithm for the composite collocation problem in space and time. We stated the underlying composite collocation problem in matrix formulation, spanning the full domain in space and time, and decomposed it into three layers: spatial decomposition, time-stepping and quadrature nodes. With suitable transformations, we could show the similarity of PFASST's iteration matrix to a block-diagonal matrix, containing either $N/2$ time-collocation blocks of size $2ML$ or $NL/2$ collocation blocks of size $2M$. While in the first case the analysis is rigorous, in the second case periodicity in time is assumed.

We identified 4 different strategies to test the convergence properties of PFASST using the block diagonalization of the iteration matrix. Along the lines of two prototype problems, we investigated the quality of the predictions given by these strategies compared to the numerical results from PFASST. We explored the effect of PFASST on different modes of the solution, depending on the initial values.

With a suitable measure for the convergence speed of PFASST at hand, the central next step would be to estimate the parallel performance of this algorithm in comparison to serial SDC runs. To this end, block diagonalizations of PFASST and SDC can be compared following the strategies presented in this work. This would augment the current speedup considerations of PFASST as stated in [16] by providing estimates for the actual iterations counts. In addition, we have identified the following topics as relevant for further studies.

Detailed parameter and component studies. So far, we have only investigated simple 1D problems, demonstrating how the LFA of the iteration matrix can be used to predict the convergence

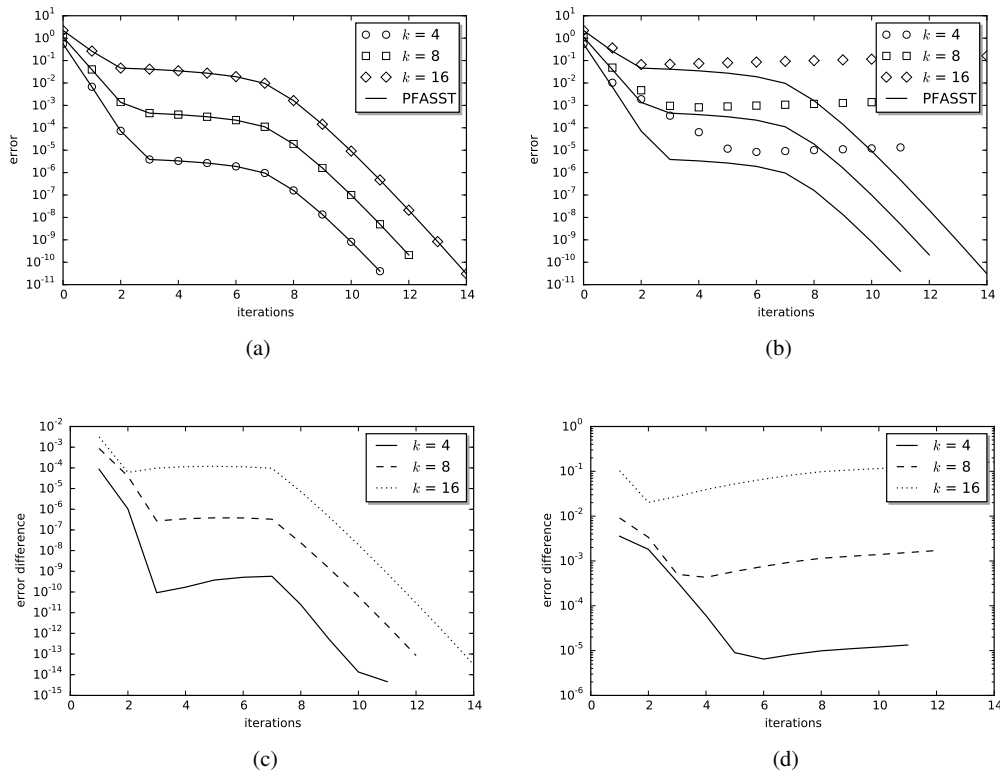


Figure 5. Strategy 4 for various k , using time collocation blocks on the left and collocation blocks on the right for the advection problem. In the second row the difference between the error and the error prediction is plotted.

behavior of PFASST for different situations. These examples can serve as a blueprint for a much deeper and more detailed analysis of PFASST's convergence properties for various problems. Also, the matrix formulation of PFASST allows us to exchange parts more easily. We can test other smoothers than SDC, change the quadrature rules used on the subintervals, vary interpolation and restriction on space (and even time) and apply iterative solvers like standard multigrid in space for inverting the spatial operators.

Non-linear functions. We restricted our self to linear problems in order to apply the Local Fourier Analysis. However, the notation used is derived from the Full Approximation Scheme and therefore is also applicable to non-linear right-hand sides. Meaning that we use a non-linear function $f(\mathbf{U}(\tau))$ of $\mathbf{A}\mathbf{U}(\tau)$, also meaning that most matrices are exchanged by operators. These changes make a convergence analysis more difficult.

Extension to multiple levels. In contrast to Parareal, the PFASST algorithm is designed to use more than two levels. Due to the simplification of the notation and the rigor of the argumentation chain, this fact was not exploited. For the same reasons, the interpolation and restriction matrices effected only the spatial dimension, although it is possible to construct coarse levels with less quadrature nodes than on the fine level. The effects on the formalism in Section 4 would be minor. It is another story, if a coarse level is constructed where two or more subintervals from the fine level are merged to one. This would be a step in the direction of full Space-Time MultiGrid, but some work is needed to adjust the formalism in Section 4 for a similar convergence analysis. Another step towards full ST-MG would be the use the exact solution on the coarsest level instead of one or

more SDC sweeps, but first brief experiments showed no significant difference between the use of the exact solution or the use of SDC Sweeps.

Rigorous convergence analysis. The usual attempt in Multigrid theory for a rigorous convergence analysis contains the proof of the smoothing and approximation property. Both endeavors are difficult on their own, but, in our case, are further impeded by the matrices \mathbf{Q} , \mathbf{Q}_Δ . These matrices are dense and yield no obvious structural properties, which could be exploited. First steps towards a more rigorous analysis would be to resolve this problem.

REFERENCES

1. Burrage K. Parallel methods for ODEs. *Advances in Computational Mathematics* 1997; **7**:1–3. URL <http://dx.doi.org/10.1023/A:1018997130884>.
2. Iserles A, Nørsett S. On the theory of parallel Runge-Kutta methods. *IMA Journal of numerical Analysis* 1990; **10**(4):463–488.
3. Butcher J. Order and stability of parallel methods for stiff problems. *Advances in Computational Mathematics* 1997; **7**(1):79–96.
4. Christlieb AJ, Macdonald CB, Ong BW. Parallel high-order integrators. *SIAM Journal on Scientific Computing* 2010; **32**(2):818–835. URL <http://dx.doi.org/10.1137/09075740X>.
5. Güttel S. A parallel overlapping time-domain decomposition method for ODE's. *Domain Decomposition Methods in Science and Engineering XX*. Springer, 2013; 459–466.
6. Maday Y, Rønquist EM. Parallelization in time through tensor-product space-time solvers. *Comptes Rendus Mathématique* 2008; **346**(1–2):113–118.
7. Sheen D, Sloan IH, Thomée V. A parallel method for time discretization of parabolic equations based on Laplace transformation and quadrature. *IMA Journal of Numerical Analysis* 2003; **23**(2):269–299.
8. Gander MJ. A waveform relaxation algorithm with overlapping splitting for reaction diffusion equations. *Numerical linear algebra with applications* 1999; **6**(2):125–145.
9. Vandewalle S, Roose D. The parallel waveform relaxation multigrid method. *Parallel Processing for Scientific Computing* 1989; :152–156.
10. Nievergelt J. Parallel methods for integrating ordinary differential equations. *Commun. ACM* 1964; **7**(12):731–733. URL <http://dx.doi.org/10.1145/355588.365137>.
11. Chartier P, Philippe B. A parallel shooting technique for solving dissipative ODE's. *Computing* 1993; **51**(3–4):209–236. URL <http://dx.doi.org/10.1007/BF02238534>.
12. Bellen A, Zennaro M. Parallel algorithms for initial-value problems for difference and differential equations. *Journal of Computational and Applied Mathematics* 1989; **25**(3):341–350. URL [http://dx.doi.org/10.1016/0377-0427\(89\)90037-X](http://dx.doi.org/10.1016/0377-0427(89)90037-X).
13. Lions JL, Maday Y, Turinici G. A "parareal" in time discretization of PDE's. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics* 2001; **332**:661–668. URL [http://dx.doi.org/10.1016/S0764-4442\(00\)01793-6](http://dx.doi.org/10.1016/S0764-4442(00)01793-6).
14. Gander M, Jiang YL, Li RJ. Parareal Schwarz waveform relaxation methods. *Domain Decomposition Methods in Science and Engineering XX, Lecture Notes in Computational Science and Engineering*, vol. 91, Bank R, Holst M, Widlund O, Xu J (eds.). Springer Berlin Heidelberg, 2013; 451–458. URL http://dx.doi.org/10.1007/978-3-642-35275-1_53.
15. Minion ML. A hybrid parareal spectral deferred corrections method. *Communications in Applied Mathematics and Computational Science* 2010; **5**(2):265–301. URL <http://dx.doi.org/10.2140/camcos.2010.5.265>.
16. Emmett M, Minion ML. Toward an efficient parallel in time method for partial differential equations. *Communications in Applied Mathematics and Computational Science* 2012; **7**:105–132. URL <http://dx.doi.org/10.2140/camcos.2012.7.105>.
17. Trottenberg U, Oosterlee CW, Schuller A. *Multigrid*. Academic press, 2000.
18. Hackbusch W. Parabolic multigrid methods. *Computing Methods in Applied Sciences and Engineering, VI* 1984; :189–197. URL <http://dl.acm.org/citation.cfm?id=4673.4714>.
19. Lubich C, Ostermann A. Multi-grid dynamic iteration for parabolic equations. *BIT Numerical Mathematics* 1987; **27**(2):216–234. URL <http://dx.doi.org/10.1007/BF01934186>.
20. Vandewalle S, Van de Velde E. Space-time concurrent multigrid waveform relaxation. *Annals of Numer. Math* 1994; **1**:347–363.
21. Falgout R, Friedhoff S, Kolev TV, MacLachlan S, Schroder JB. Parallel time integration with multigrid. *SIAM Journal on Scientific Computing* 2014; **36**(6):C635–C661.
22. Horton G, Vandewalle S. A space-time multigrid method for parabolic partial differential equations. *SIAM Journal on Scientific Computing* 1995; **16**(4):848–864. URL <http://dx.doi.org/10.1137/0916050>.
23. Gander MJ, Neumüller M. Analysis of a new space-time parallel multigrid algorithm for parabolic problems. *arXiv preprint arXiv:1411.0519* 2014; .
24. Gander MJ, Vandewalle S. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing* 2007; **29**(2):556–578. URL <http://dx.doi.org/10.1137/05064607X>.
25. Dutt A, Greengard L, Rokhlin V. Spectral deferred correction methods for ordinary differential equations. *BIT Numerical Mathematics* 2000; **40**(2):241–266. URL <http://dx.doi.org/10.1023/A:1022338906936>.

26. Frank R, Ueberhuber CW. Iterated defect correction for the efficient solution of stiff systems of ordinary differential equations. *BIT Numerical Mathematics* 1977; **17**(2):146–159. URL <http://dx.doi.org/10.1007/BF01932286>.
27. Huang J, Jia J, Minion M. Accelerating the convergence of spectral deferred correction methods. *Journal of Computational Physics* 2006; **214**(2):633 – 656. URL <http://dx.doi.org/10.1016/j.jcp.2005.10.004>.
28. Layton AT, Minion ML. Conservative multi-implicit spectral deferred correction methods for reacting gas dynamics. *Journal of Computational Physics* 2004; **194**(2):697 – 715. URL <http://dx.doi.org/10.1016/j.jcp.2003.09.010>.
29. Minion ML. Semi-implicit projection methods for incompressible flow based on spectral deferred corrections. *Applied Numerical Mathematics* 2004; **48**(3–4):369 – 387. URL <http://dx.doi.org/10.1016/j.apnum.2003.11.005>, Workshop on Innovative Time Integrators for PDEs.
30. Bourlioux A, Layton AT, Minion ML. High-order multi-implicit spectral deferred correction methods for problems of reactive flow. *Journal of Computational Physics* 2003; **189**(2):651 – 675. URL [http://dx.doi.org/10.1016/S0021-9991\(03\)00251-1](http://dx.doi.org/10.1016/S0021-9991(03)00251-1).
31. Guibert D, Tromeur-Dervout D. Parallel deferred correction method for CFD problems. *Parallel Computational Fluid Dynamics* 2006, Kwon J, Ecer A, Satofuka N, Periaux J, Fox P (eds.). Elsevier Science B.V.: Amsterdam, 2007; 131 – 138. URL <http://dx.doi.org/10.1016/B978-044453035-6/50019-5>.
32. Minion ML, Williams SA. Parareal and spectral deferred corrections. *AIP Conference Proceedings*, vol. 1048, 2008; 388. URL <http://link.aip.org/link/doi/10.1063/1.2990941>.
33. Weiser M. Faster SDC convergence on non-equidistant grids with DIRK sweeps 2013. URL <http://opus4.kobv.de/opus4-zib/files/1866/ZR-13-30.pdf>, ZIB Report 13–30.
34. Winkel M, Speck R, Ruprecht D. A high-order Boris integrator. *Journal of computational physics* 2015; **295**:456–474.
35. Speck R, Ruprecht D, Emmett M, Minion M, Bolten M, Krause R. A multi-level spectral deferred correction method. *BIT Numerical Mathematics* 2015; **55**(3):843–867.
36. Koehler F. Pfasst tikz. <https://github.com/Parallel-in-Time/pfasst-tikz> 2015.
37. Friedhoff S, MacLachlan S. A generalized predictive analysis tool for multigrid methods. *Numerical Linear Algebra with Applications* 2015; **22**(4):618–647, doi:10.1002/nla.1977. URL <http://dx.doi.org/10.1002/nla.1977>, nla.1977.
38. Kelley C T. Iterative methods for linear and nonlinear equations. *Raleigh N. C.: North Carolina State University* 1995; .